(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 119653109 A
(43)申请公布日 2025.03.18

(21)申请号 202510124181.7

(22)申请日 2020.11.09

(30)优先权数据
62/933,971 2019.11.11 US
62/933,954 2019.11.11 US

(62)分案原申请数据
202080092295.3 2020.11.09

(71)申请人 LG电子株式会社
地址 韩国首尔

(72)发明人 具文模 金昇焕 M·萨勒海法尔 林宰显

(74)专利代理机构 北京三友知识产权代理有限公司 11127
专利代理师 孙东喜

(51)Int.Cl.
H04N 19/60(2014.01)
H04N 19/122(2014.01)
H04N 19/129(2014.01)
H04N 19/132(2014.01)
H04N 19/176(2014.01)
H04N 19/18(2014.01)
H04N 19/186(2014.01)
H04N 19/46(2014.01)
H04N 19/70(2014.01)

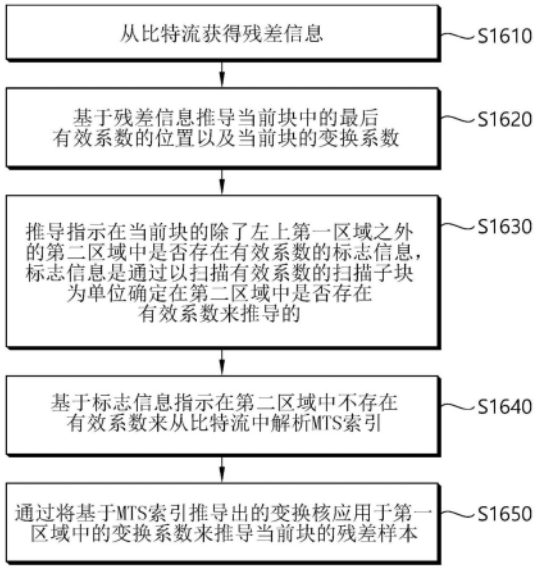
权利要求书2页 说明书49页 附图15页

(54)发明名称

图像信息的解码设备、编码设备、发送设备和存储介质

(57)摘要

本公开提供了图像信息的解码设备、编码设备、发送设备和存储介质。根据本文档的图像解码方法包括以下步骤:推导指示在当前块的除了左上角的第一区域之外的第二区域中是否存在有效系数的标志信息;基于标志信息指示在第二区域中不存在有效系数,从比特流中解析MTS索引;以及通过将基于MTS索引推导的变换核应用于第一区域的变换系数来推导当前块的残差样本,其中,标志信息可以是通过针对扫描有效系数的每个扫描子块确定在第二区域中是否存在有效系数而推导的。



CN 119653109 A

1.一种用于对图像信息进行解码的设备,所述设备包括:

存储器;以及

与所述存储器联接的至少一个处理器,所述至少一个处理器被配置为:

从比特流获得残差信息;

基于所述残差信息推导当前块中的最后有效系数的位置;

推导指示在所述当前块的除了左上第一区域之外的第二区域中是否存在有效系数的标志信息;

推导用于所述当前块的变换系数;

基于所述标志信息指示在所述第二区域中不存在有效系数,从所述比特流中解析索引;并且

通过基于所述索引将逆变换应用于所述第一区域中的变换系数来推导用于所述当前块的残差样本,

其中,所述至少一个处理器被配置为:

通过根据逆对角扫描顺序以预设的扫描子块为单位对扫描子块进行扫描来解析与所述扫描子块中是否存在有效系数相关的标志;

确定所述扫描子块是否属于所述第二区域;并且

确定所述当前块的颜色索引是否指示亮度分量,并且

其中,基于所述标志的值为1、所述扫描子块属于所述第二区域、并且所述颜色索引指示所述亮度分量,所述标志信息指定在所述第二区域中存在所述有效系数。

2.一种用于对图像信息进行编码的设备,所述设备包括:

存储器;以及

与所述存储器联接的至少一个处理器,所述至少一个处理器被配置为:

推导用于当前块的预测样本;

基于所述预测样本推导用于所述当前块的残差样本;

基于针对所述残差样本的变换推导用于所述当前块的变换系数;

构造图像信息来推导指示在所述当前块的除了左上第一区域之外的第二区域中是否存在有效系数的标志信息;并且

基于所述标志信息指示在所述第二区域中不存在有效系数,对索引和基于用于所述当前块的所述变换系数推导的残差信息进行编码,

其中,所述至少一个处理器被配置为:

通过根据逆对角扫描顺序以预设的扫描子块为单位对扫描子块进行扫描来解析与所述扫描子块中是否存在所述有效系数相关的标志;

确定所述扫描子块是否属于所述第二区域;并且

确定所述当前块的颜色索引是否指示亮度分量,并且

其中,基于所述标志的值为1、所述扫描子块属于所述第二区域、并且所述颜色索引指示所述亮度分量,所述标志信息指定在所述第二区域中存在所述有效系数。

3.一种用于发送用于图像信息的数据的设备,所述设备包括:

至少一个处理器,所述至少一个处理器被配置为生成用于图像的比特流,其中,所述比特流是基于以下操作生成的:推导用于当前块的预测样本;基于所述预测样本推导用于所

述当前块的残差样本;基于针对所述残差样本的变换推导所述用于当前块的变换系数;构造图像信息来推导指示在所述当前块的除了左上第一区域之外的第二区域中是否存在有效系数的标志信息;以及基于所述标志信息指示在所述第二区域中不存在有效系数,对索引和基于用于所述当前块的所述变换系数推导的残差信息进行编码以生成所述比特流;以及

发送器,所述发送器被配置为发送包括所述比特流的所述数据,

其中,所述至少一个处理器被配置为:

通过根据逆对角扫描顺序以预设的扫描子块为单位对扫描子块进行扫描来解析与所述扫描子块中是否存在所述有效系数相关的标志;

确定所述扫描子块是否属于所述第二区域;并且

确定所述当前块的颜色索引是否指示亮度分量,并且

其中,基于所述标志的值为1、所述扫描子块属于所述第二区域、并且所述颜色索引指示所述亮度分量,所述标志信息指定在所述第二区域中存在所述有效系数。

4.一种非暂时性计算机可读数字存储介质,所述非暂时性计算机可读数字存储介质存储有通过一种方法生成的比特流,所述方法包括以下步骤:

推导用于当前块的预测样本;

基于所述预测样本推导用于所述当前块的残差样本;

基于针对所述残差样本的变换推导用于所述当前块的变换系数;

构造图像信息来推导指示在所述当前块的除了左上第一区域之外的第二区域中是否存在有效系数的标志信息;以及

基于所述标志信息指示在所述第二区域中不存在有效系数,对索引和基于用于所述当前块的所述变换系数推导的残差信息进行编码,

其中,构造所述图像信息的步骤包括以下步骤:

通过根据逆对角扫描顺序以预设的扫描子块为单位对扫描子块进行扫描来解析与所述扫描子块中是否存在所述有效系数相关的标志;

确定所述扫描子块是否属于所述第二区域;以及

确定所述当前块的颜色索引是否指示亮度分量,并且

其中,基于所述标志的值为1、所述扫描子块属于所述第二区域、并且所述颜色索引指示所述亮度分量,所述标志信息指定在所述第二区域中存在所述有效系数。

图像信息的解码设备、编码设备、发送设备和存储介质

[0001] 本申请是原案申请号为202080092295.3的发明专利申请(国际申请号:PCT/KR2020/015588,申请日:2020年11月9日,发明名称:基于变换的图像编码方法及其装置)的分案申请。

技术领域

[0002] 本公开总体涉及一种图像编码技术,并且更具体地,涉及图像编码系统中的基于变换的图像编码方法及其设备。

背景技术

[0003] 如今,在各个领域中对诸如4K、8K或更高的超高清(UHD)图像/视频这样的高分辨率和高质量图像/视频的需求已经不断增长。随着图像/视频数据变成更高分辨率和更高质量,与传统图像数据相比,所发送的信息量或比特量增加。因此,当使用诸如传统有线/无线宽带线这样的介质来发送图像数据或者使用现有存储介质来存储图像/视频数据时,其传输成本和存储成本增加。

[0004] 另外,如今,对于诸如虚拟现实(VR)和人工现实(AR)内容或全息图等这样的沉浸式媒体的兴趣和需求正在增加,并且对具有与诸如游戏图像这样的真实图像不同的图像特征的图像/视频的广播正在增加。

[0005] 因此,需要有效地压缩并发送或存储以及再现具有如上所述的各种特征的高分辨率和高质量图像/视频的信息的高效图像/视频压缩技术。

发明内容

[0006] 技术目的

[0007] 本公开的一技术方面在于提供用于增加图像编码效率的方法和设备。

[0008] 本公开的另一技术方面在于提供用于增加编码变换索引的效率的方法和设备。

[0009] 本公开的又一技术方面在于提供使用MTS的图像编码方法和图像编码设备。

[0010] 本公开的再一技术方面在于提供用于发信号通知MTS索引的的图像编码方法和图像编码设备。

[0011] 技术方案

[0012] 根据本公开的实施方式,提供了一种由解码设备执行的图像解码方法。该方法可以包括:推导指示在当前块的除了左上第一区域之外的第二区域中是否存在有效系数的标志信息;基于标志信息指示在第二区域中不存在有效系数,从比特流中解析MTS索引;以及通过将基于MTS索引推导的变换核应用于第一区域中的变换系数来推导当前块的残差样本,其中,标志信息可以是通过以扫描有效系数的扫描子块为单位确定在第二区域中是否存在有效系数而推导的。

[0013] 当指示在扫描子块中是否存在有效系数的标志的值为1并且在扫描子块中存在的有效系数位于第二区域中时,标志信息可以指示在第二区域中存在有效系数。

[0014] 当指示在扫描子块中是否存在有效系数的标志的值为1并且扫描子块位于第二区域中时,标志信息可以指示在第二区域中存在有效系数。

[0015] 标志信息可以初始被设置为1,并且当有效系数位于第二区域时,标志信息可以被改变为0。

[0016] 第一区域可以是当前块的左上16×16区域,并且扫描子块可以是4×4块。

[0017] 可以从当前块中的最后有效系数的位置开始根据逆对角扫描方向对扫描子块进行扫描。

[0018] 根据本公开的实施方式,提供了一种由编码设备执行的图像编码方法。该方法可以包括:基于残差样本的一次变换推导当前块的变换系数;对当前块的除了左上第一区域之外的第二区域进行清零;构造图像信息来以扫描有效系数的扫描子块为单位推导指示在第二区域中是否存在有效系数的标志信息;以及基于标志信息指示在第二区域中不存在有效系数,对MTS索引和基于当前块的变换系数推导的残差信息进行编码。

[0019] 根据本公开的又一实施方式,可以提供一种数字存储介质,该数字存储介质存储包括根据由编码设备执行的图像编码方法生成的比特流和编码的图像信息的图像数据。

[0020] 根据本公开的又一实施方式,可以提供一种数字存储介质,该数字存储介质存储包括编码的图像信息和比特流的图像数据以使解码设备执行图像解码方法。

[0021] 技术效果

[0022] 根据本公开,可以增加整体图像/视频压缩效率。

[0023] 根据本公开,可以增加编码MTS索引的效率。

[0024] 根据本公开,可以增加使用MTS的图像编码方法的效率。

[0025] 可以通过本公开的具体示例获得的效果不限于上面列出的效果。例如,可以存在相关领域的普通技术人员可以理解或从本公开推导出的各种技术效果。因此,本公开的特定效果不限于本公开中明确描述的那些,并且可以包括可以根据本公开的技术特征理解或推导出的各种效果。

附图说明

[0026] 图1示意性地例示了可应用本公开的视频/图像编码系统的示例。

[0027] 图2是示意性地例示可应用本公开的视频/图像编码设备的配置的图。

[0028] 图3是示意性地例示可应用本公开的视频/图像解码设备的配置的图。

[0029] 图4例示了应用本公开的内容流系统的结构。

[0030] 图5示意性地例示了根据本公开的实施方式的多变换技术。

[0031] 图6示意性地示出了65个预测方向的帧内定向模式。

[0032] 图7是用于说明根据本公开的实施方式的RST的图。

[0033] 图8是例示根据示例的将正向一次变换的输出数据布置成一维向量的顺序的图。

[0034] 图9是例示根据示例的将正向二次变换的输出数据布置成二维块的顺序的图。

[0035] 图10是例示根据本文档的实施方式的广角帧内预测模式的图。

[0036] 图11是例示LFNST被应用于的块形状的图。

[0037] 图12是例示根据示例的正向LFNST的输出数据的布置的图。

[0038] 图13例示了根据示例的在应用了4×4LFNST的块中的清零。

[0039] 图14例示了根据示例的在应用了8×8LFNST的块中的清零。

[0040] 图15例示了根据本公开的实施方式的32×32变换块的扫描。

[0041] 图16是例示根据本公开的实施方式的视频解码设备的操作的流程图。

[0042] 图17是例示根据本公开的实施方式的视频编码设备的操作的流程图。

具体实施方式

[0043] 尽管本公开可能易于进行各种修改并且包括各种实施方式,但是其具体实施方式已在附图中通过示例的方式示出,并且现在将对其进行详细描述。然而,这并不旨在将本公开限制于本文公开的具体实施方式。本文中使用的术语只是出于描述特定实施方式的目的,并不旨在限制本公开的技术思路。除非上下文另外清楚指示,否则单数形式可以包括复数形式。诸如“包括”和“具有”之类的术语旨在指示存在以下描述中使用的特征、数字、步骤、操作、元件、组件或其组合,因此不应被理解为预先排除了存在或添加一个或更多个不同特征、数字、步骤、操作、元件、组件或其组合的可能性。

[0044] 此外,为了方便描述彼此不同的特性功能,独立地例示了本文中描述的附图上的各组件,然而,并不意指各组件由单独的硬件或软件来实现。例如,可以组合这些组件中的任何两个或更多个以形成单个组件,并且任何单个组件可以被划分成多个组件。其中组件被组合和/或划分的实施方式将属于本公开的专利权的范围,只要它们不脱离本公开的实质即可。

[0045] 在下文中,将参照附图更详细地说明本公开的优选实施方式。另外,在附图中,相同的附图标号用于相同的组件,并且将省略对相同组件的重复描述。

[0046] 本文档涉及视频/图像编码。例如,本文档中公开的方法/示例可以涉及VVC(通用视频编码)标准(ITU-T Rec.H.266)、VVC之后的下一代视频/图像编码标准、或其它视频编码相关标准(例如,HEVC(高效视频编码)标准(ITU-T Rec.H.265)、EVC(基本视频编码)标准、AVS2标准等)。

[0047] 在本文档中,可以提供与视频/图像编码相关的各种实施方式,并且除非相反地指定,否则这些实施方式可以彼此组合并执行。

[0048] 在本文档中,视频可以是指一段时间内一系列图像的集合。通常,图片是指表示特定时间区域的图像的单元,而条带/贴片是构成图片的一部分的单元。条带/贴片可以包括一个或更多个编码树单元(CTU)。一幅图片可以由一个或更多个条带/贴片组成。一幅图片可以由一个或多个贴片组组成。一个贴片组可以包括一个或更多个贴片。

[0049] 像素或图元(pel)可以是指构成一幅图片(或图像)的最小单元。另外,“样本”可用作与像素相对应的术语。样本通常可以表示像素或像素值,并且可以仅表示亮度分量的像素/像素值或仅表示色度分量的像素/像素值。另选地,样本可以意指空间域中的像素值,或者当该像素值被变换为频域时,其可以意指频域中的变换系数。

[0050] 单元可以表示图像处理的基本单位。单元可以包括特定区域和与该区域相关的信息中的至少一个。一个单元可以包括一个亮度块和两个色度(例如,cb、cr)块。根据情况,可以将单元和诸如块、区域等这样的术语互换地使用。在通常情况下,M×N块可以包括由M列和N行组成的样本(或样本阵列)或变换系数的集(或阵列)。

[0051] 在本文档中,术语“/”和“,”应该被解释为指示“和/或”。例如,表述“A/B”可以意指

“A和/或B”。另外,“A、B”可以意指“A和/或B”。另外,“A/B/C”可以意指“A、B和/或C中的至少一个”。另外,“A/B/C”可以意指“A、B和/或C中的至少一个”。

[0052] 另外,在该文档中,术语“或”应该被解释为指示“和/或”。例如,表述“A或B”可以包括1)仅A、2)仅B和/或3)A和B二者。换句话说,本文档中的术语“或”应该被解释为指示“附加地或另选地”。

[0053] 在本公开中,“A和B中的至少一个”可以意指“仅A”、“仅B”或“A和B两者”。此外,在本公开中,表述“A或B中的至少一个”或者“A和/或B中的至少一个”可以被解释为“A和B中的至少一个”。

[0054] 此外,在本公开中,“A、B和C中的至少一个”可以意指“仅A”、“仅B”、“仅C”或“A、B和C的任何组合”。此外,“A、B或C中的至少一个”或“A、B和/或C中的至少一个”可以意指“A、B和C中的至少一个”。

[0055] 另外,本公开中使用的括号可以表示“例如”。具体地,当指示为“预测(帧内预测)”时,可以意指“帧内预测”被提出为“预测”的示例。换句话说,本公开的“预测”不限于“帧内预测”,并且“帧内预测”是作为“预测”的示例而提出的。另外,当指示为“预测(即,帧内预测)”时,这也可以意指“帧内预测”被提出为“预测”的示例。

[0056] 在本公开中的一个附图中单独描述的技术特征可以单独地实现或可以同时实现。

[0057] 图1示意性地例示了可应用本公开的视频/图像编码系统的示例。

[0058] 参照图1,视频/图像编码系统可以包括第一装置(源装置)和第二装置(接收装置)。源装置可以经由数字存储介质或网络以文件或流的形式将编码后的视频/图像信息或数据传递到接收装置。

[0059] 源装置可以包括视频源、编码设备和发送器。接收装置可以包括接收器、解码设备和渲染器。编码设备可以被称为视频/图像编码设备,并且解码设备可以被称为视频/图像解码设备。发送器可以被包括在编码设备中。接收器可以被包括在解码设备中。渲染器可以包括显示器,并且显示器可以被配置为单独的装置或外部组件。

[0060] 视频源可以通过捕获、合成或生成视频/图像的处理来获得视频/图像。视频源可以包括视频/图像捕获装置和/或视频/图像生成装置。视频/图像捕获装置可以包括例如一个或更多个相机、包括先前捕获的视频/图像的视频/图像档案等。视频/图像生成装置可以包括例如计算机、平板计算机和智能电话,并且可以(电子地)生成视频/图像。例如,可以通过计算机等生成虚拟视频/图像。在这种情况下,视频/图像捕获处理可以被生成相关数据的处理取代。

[0061] 编码设备可以对输入视频/图像进行编码。编码设备可以执行诸如针对压缩和编码效率的预测、变换和量化这样的一系列过程。编码后的数据(编码后的视频/图像信息)可以以比特流的形式输出。

[0062] 发送器可以通过数字存储介质或网络以文件或流的形式将以比特流的形式输出的编码后的视频/图像信息或数据发送到接收装置的接收器。数字存储介质可以包括诸如USB、SD、CD、DVD、蓝光、HDD、SSD等这样的各种存储介质。发送器可以包括用于通过预定文件格式生成媒体文件的元件,并且可以包括用于通过广播/通信网络进行发送的元件。接收器可以接收/提取比特流,并且将接收/提取的比特流发送到解码设备。

[0063] 解码设备可以通过执行与编码设备的操作对应的诸如解量化、逆变换、预测等这

样的一系列过程来解码视频/图像。

[0064] 渲染器可以渲染解码后的视频/图像。可以通过显示器显示渲染后的视频/图像。

[0065] 图2是示意性地例示可应用本公开的视频/图像编码设备的配置的图。在下文中，所谓的视频编码设备可以包括图像编码设备。

[0066] 参照图2，编码设备200可以包括图像划分器210、预测器220、残差处理器230、熵编码器240、加法器250、滤波器260和存储器270。预测器220可以包括帧间预测器221和帧内预测器222。残差处理器230可以包括变换器232、量化器233、解量化器234、逆变换器235。残差处理器230可以进一步包括减法器231。加法器250可以称为重构器或重构块生成器。根据实施方式，以上描述的图像划分器210、预测器220、残差处理器230、熵编码器240、加法器250和滤波器260可以由一个或更多个硬件组件(例如，编码器芯片组或处理器)构成。此外，存储器270可以包括解码图片缓冲器(DPB)，并且可以由数字存储介质构成。硬件组件可以进一步包括作为内部/外部组件的存储器270。

[0067] 图像划分器210可以将输入到编码设备200的输入图像(或图片或帧)划分为一个或更多个处理单元。作为一个示例，处理单元可以被称为编码单元(CU)。在这种情况下，从编码树单元(CTU)或最大编码单元(LCU)开始，可以根据四叉树二叉树三叉树(QTBTTT)结构来递归地划分编码单元。例如，基于四叉树结构、二叉树结构和/或三叉树结构，一个编码单元可以被划分为深度较深的多个编码单元。在这种情况下，例如，可以首先应用四叉树结构，并且可以稍后应用二叉树结构和/或三叉树结构。另选地，可以首先应用二叉树结构。可以基于未进一步划分的最终编码单元来执行根据本公开的编码过程。在这种情况下，基于根据图像特性的编码效率，最大编码单元可以直接用作最终编码单元。另选地，可以根据需要将编码单元递归地划分为更深的深度的编码单元，由此可以将最佳大小的编码单元用作最终编码单元。这里，编码过程可以包括诸如预测、变换和重构之类的过程，其将在后面描述。作为另一示例，处理单元可以进一步包括预测单元(PU)或变换单元(TU)。在这种情况下，预测单元和变换单元可以与上述最终编码单元分开或划分开。预测单元可以是样本预测的单元，并且变换单元可以是用于推导变换系数的单元和/或用于从变换系数推导残差信号的单元。

[0068] 根据情况，可以使用单元和诸如块、区域等之类的术语来代替彼此。在通常情况下，M×N块可以表示由M列和N行构成的样本或变换系数的集。样本通常可以表示像素或像素值，并且可以仅表示亮度分量的像素/像素值，或者仅表示色度分量的像素/像素值。样本可用作与一幅图片(或图像)的像素或图元(pel)相对应的术语。

[0069] 减法器231从输入图像信号(原始块、原始样本阵列)中减去从预测器220输出的预测信号(预测块、预测样本阵列)以生成残差信号(残差块、残差样本阵列)，并且所生成的残差信号被发送到变换器232。预测器220可以对处理目标块(下文称为“当前块”)执行预测，并且可以生成包括当前块的预测样本的预测块。预测器220可以确定是以当前块或CU为基础应用帧内预测还是应用帧间预测。如稍后在每个预测模式的描述中所讨论的，预测器可以生成诸如预测模式信息之类的与预测有关的各种信息，并将所生成的信息发送到熵编码器240。关于预测的信息可以在熵编码器240中编码并以比特流的形式输出。

[0070] 帧内预测器222可以通过参考当前图片中的样本来预测当前块。根据预测模式，参考样本可以位于当前块的附近或与当前块分开。在帧内预测中，预测模式可以包括多种非

定向模式和多种定向模式。非定向模式可以包括例如DC模式和平面模式。根据预测方向的详细程度,定向模式可以包括例如33种定向预测模式或65种定向预测模式。然而,这仅仅是示例,并且根据设置,可以使用更多或更少的定向预测模式。帧内预测器222可以通过使用应用于邻近块的预测模式来确定应用于当前块的预测模式。

[0071] 帧间预测器221可以基于参照图片上的运动向量所指定的参考块(参考样本阵列)来推导针对当前块的预测块。此时,为了减少在帧间预测模式下发送的运动信息的量,可以基于邻近块与当前块之间的运动信息的相关性以块、子块或样本为基础来预测运动信息。运动信息可以包括运动向量和参照图片索引。运动信息还可以包括帧间预测方向(L0预测、L1预测、Bi预测等)信息。在帧间预测的情况下,邻近块可以包括当前图片中存在的空间邻近块和参照图片中存在的时间邻近块。包括参考块的参照图片和包括时间邻近块的参照图片可以彼此相同或彼此不同。时间邻近块可以被称为并置参考块、并置CU(colCU)等,并且包括时间邻近块的参照图片可以被称为并置图片(colPic)。例如,帧间预测器221可以基于邻近块来配置运动信息候选列表,并且生成指示哪个候选被用于推导当前块的运动向量和/或参照图片索引的信息。可以基于各种预测模式来执行帧间预测。例如,在跳变模式和合并模式的情况下,帧间预测器221可以使用邻近块的运动信息作为当前块的运动信息。在跳变模式下,与合并模式不同,不能发送残差信号。在运动信息预测(运动向量预测、MVP)模式的情况下,邻近块的运动向量可以被用作运动向量预测符,并且可以通过发信号通知运动向量差来指示当前块的运动向量。

[0072] 预测器220可以基于各种预测方法来生成预测信号。例如,预测器可以将帧内预测或帧间预测应用于对一个块的预测,并且也可以同时应用帧内预测和帧间预测。这可以被称为组合的帧间和帧内预测(CIIP)。另外,预测器可以基于块内复制(IBC)预测模式或调色板模式,以便对块执行预测。IBC预测模式或调色板模式可以被用于诸如屏幕内容编码(SCC)这样的游戏等的内容图像/视频编码。尽管IBC基本上在当前块中执行预测,但是其执行方式与帧间预测的相似之处在于,它推导当前块中的参考块。也就是说,IBC可以使用本公开中描述的帧间预测技术中的至少一种。

[0073] 通过帧间预测器221和/或帧内预测器222生成的预测信号可以被用于生成重构信号或者生成残差信号。变换器232可以通过向残差信号应用变换技术来生成变换系数。例如,变换技术可以包括离散余弦变换(DCT)、离散正弦变换(DST)、Karhunen-Loève变换(KLT)、基于图的变换(GBT)或有条件非线性变换(CNT)中的至少一种。这里,GBT意指当用曲线图表示像素之间的关系信息时从曲线图获得的变换。CNT是指基于使用所有先前重构的像素生成的预测信号而获得的变换。另外,变换处理可以被应用于大小相同的正方形像素块,或者可以应用于大小可变的块而非正方形的块。

[0074] 量化器233可以对变换系数进行量化并且将它们发送到熵编码器240,并且熵编码器240可以对量化后的信号(关于量化变换系数的信息)进行编码并且输出比特流中的编码后的信号。关于量化变换系数的信息可以被称为残差信息。量化器233可以基于系数扫描顺序将块类型的量化变换系数重新布置成一维向量形式,并且基于一维向量形式的量化变换系数来生成关于量化变换系数的信息。熵编码器240可以执行诸如例如指数哥伦布(exponential Golomb)、上下文自适应可变长度编码(CAVLC)、上下文自适应二进制算术编码(CABAC)等这样的各种编码方法。熵编码器240可以对除了量化变换系数(例如,语法元素

的值等)之外的视频/图像重构所需的信息一起或分别进行编码。编码后的信息(例如,编码后的视频/图像信息)可以以比特流的形式在网络抽象层(NAL)的单元基础上进行发送或存储。视频/图像信息还可以包括关于诸如适应参数集(APS)、图片参数集(PPS)、序列参数集(SPS)、视频参数集(VPS)等这样的各种参数集的信息。另外,视频/图像信息还可以包括常规约束信息。在本公开中,从编码设备发送到/发信号通知给解码设备的信息和/或语法元素可以被包括在视频/图像信息中。视频/图像信息可以通过上述编码过程进行编码并且被包括在比特流中。可以通过网络传输比特流,或者将其存储在数字存储介质中。这里,网络可以包括广播网络、通信网络和/或类似物,并且数字存储介质可以包括诸如USB、SD、CD、DVD、蓝光、HDD、SSD等这样的各种存储介质。发送从熵编码器240输出的信号的发送器(未示出)或存储其的存储器(未示出)可以被配置为编码设备200的内部/外部元件,或者发送器可以被包括在熵编码器240中。

[0075] 从量化器233输出的量化变换系数可以被用于生成预测信号。例如,通过利用解量化器234和逆变换器235向量化变换系数应用解量化和逆变换,可以重构残差信号(残差块或残差样本)。加法器155将重构后的残差信号与从帧间预测器221或帧内预测器222输出的预测信号相加,使得可以生成重构信号(重构图片、重构块、重构样本阵列)。当如在应用跳变模式的情况下一样没有针对处理目标块的残差时,可以将预测块用作重构块。加法器250可以被称为重构器或重构块生成器。所生成的重构信号可以被用于目标图片中的下一处理目标块的帧内预测,并且如随后描述的,可以被用于通过滤波进行的下一图片的帧间预测。

[0076] 此外,在图片编码和/或重构处理中,可以应用具有色度缩放的亮度映射(LMCS)。

[0077] 滤波器260可以通过向重构信号应用滤波来改善主观/客观视频质量。例如,滤波器260可以通过向重构图片应用各种滤波方法来生成修改后的重构图片,并且可以将修改后的重构图片存储在存储器270中,尤其是存储在存储器270的DPB中。各种滤波方法可以包括例如解块滤波、样本自适应偏移、自适应环形滤波器、双边滤波器等。如随后在对每种滤波方法的描述中所讨论的,滤波器260可以生成与滤波相关的各种信息,并且将所生成的信息发送到熵编码器240。关于滤波的信息可以在熵编码器240中被编码并且以比特流的形式被输出。

[0078] 已发送到存储器270的修改后的重构图片可以被用作帧间预测器221中的参照图片。据此,编码设备可以避免在应用帧间预测时编码设备200和解码设备中的预测失配,并且还可以提高编码效率。

[0079] 存储器270DPB可以存储修改后的重构图片,以便使用它作为帧间预测器221中的参照图片。存储器270可以存储从中已推导(或编码了)运动信息的当前图片中的块的运动信息和/或已经重构的图片中的块的运动信息。所存储的运动信息可以被发送到帧间预测器221,以被用作邻近块的运动信息或时间邻近块的运动信息。存储器270可以存储当前图片中的重构块的重构样本,并且将它们发送到帧内预测器222。

[0080] 图3是示意性地例示可应用本公开的视频/图像解码设备的配置的图。

[0081] 参照图3,视频解码设备300可以包括熵解码器310、残差处理器320、预测器330、加法器340、滤波器350和存储器360。预测器330可以包括帧间预测器331和帧内预测器332。残差处理器320可以包括解量化器321和逆变换器322。根据实施方式,上面已描述的熵解码器310、残差处理器320、预测器330、加法器340和滤波器350可以由一个或更多个硬件组件(例

如,解码器芯片组或处理器)构成。另外,存储器360可以包括解码图片缓冲器(DPB),并且可以由数字存储介质构成。硬件组件还可以包括存储器360作为内部/外部组件。

[0082] 当输入包括视频/图像信息的比特流时,解码设备300可以与据此已在图2的编码设备中处理视频/图像信息的处理对应地重构图像。例如,解码设备300可以基于与从比特流获得的与块分割相关的信息来推导单元/块。解码设备300可以通过使用在编码设备中应用的处理单元来执行解码。因此,解码的处理单元可以是例如编码单元,可以用编码树单元或最大编码单元顺着四叉树结构、二叉树结构和/或三叉树结构对其进行分割。可以用编码单元推导一个或更多个变换单元。并且,可以通过再现器来再现通过解码设备300解码并输出的重构图像信号。

[0083] 解码设备300可以以比特流的形式接收从图2的编码设备输出的信号,并且可以通过熵解码器310对接收到的信号进行解码。例如,熵解码器310可以对比特流进行解析,以推导图像重构(或图片重构)所需的信息(例如,视频/图像信息)。视频/图像信息还可以包括关于诸如自适应参数集(APS)、图片参数集(PPS)、序列参数集(SPS)、视频参数集(VPS)等这样的各种参数集的信息。另外,视频/图像信息还可以包括常规约束信息。解码设备可以进一步基于关于参数集的信息和/或常规约束信息对图片进行解码。在本公开中,随后将描述的发信号通知/接收的信息和/或语法元素可以通过解码过程被解码并且从比特流中获得。例如,熵解码器310可以基于诸如指数哥伦布编码、CAVLC、CABAC等这样的编码方法对比特流中的信息进行解码,并且可以输出图像重构所需的语法元素的值和关于残差的变换系数的量化值。更具体地,CABAC熵解码方法可以接收与比特流中的各语法元素对应的bin,使用解码目标语法元素信息以及邻近和解码目标块的解码信息或者在先前步骤中解码的符号/bin的信息来确定上下文模型,根据所确定的上下文模型来预测bin生成概率并且对bin执行算术解码以生成与每个语法元素值对应的符号。这里,CABAC熵解码方法可以在确定上下文模型之后使用针对下一个符号/bin的上下文模型解码的符号/bin的信息来更新上下文模型。在熵解码器310中解码的信息当中的关于预测的信息可以被提供到预测器(帧间预测器331和帧内预测器332),并且在熵解码器310中已对其执行了熵解码的残差值(即,量化变换系数)和关联的参数信息可以被输入到残差处理器320。残差处理器320可以推导残差信号(残差块、残差样本、残差样本阵列)。另外,在熵解码器310中解码的信息当中的关于滤波的信息可以被提供到滤波器350。此外,接收从编码设备输出的信号的接收器(未示出)还可以将解码设备300构成为内部/外部元件,并且接收器可以是熵解码器310的组件。此外,根据本公开的解码设备可以被称为视频/图像/图片编码设备,并且解码设备可以被分为信息解码器(视频/图像/图片信息解码器)和样本解码器(视频/图像/图片样本解码器)。信息解码器可以包括熵解码器310,并且样本解码器可以包括解量化器321、逆变换器322、加法器340、滤波器350、存储器360、帧间预测器331和帧内预测器332中的至少一个。

[0084] 解量化器321可以通过对量化变换系数进行解量化来输出变换系数。解量化器321可以将量化变换系数重新布置为二维块的形式。在这种情况下,可以基于已在编码设备中执行的系数扫描的顺序来执行重新布置。解量化器321可以使用量化参数(例如,量化步长信息)对量化变换系数执行解量化,并且获得变换系数。

[0085] 逆变换器322通过对变换系数进行逆变换来获得残差信号(残差块、残差样本阵列)。

[0086] 预测器可以对当前块执行预测,并且生成包括针对当前块的预测样本的预测块。预测器可以基于从熵解码器310输出的关于预测的信息来确定向当前块应用帧内预测还是帧间预测,并且具体地可以确定帧内/帧间预测模式。

[0087] 预测器可以基于各种预测方法来生成预测信号。例如,预测器可以将帧内预测或帧间预测应用于对一个块的预测,并且也可以同时应用帧内预测和帧间预测。这可以被称为组合的帧间和帧内预测(CIIP)。另外,预测器可以针对块的预测执行块内复制(IBC)。块内复制可以被用于诸如屏幕内容编码(SCC)这样的游戏等的内容图像/视频编码。尽管IBC基本上在当前块中执行预测,但是其执行方式与帧间预测的相似之处在于,它推导当前块中的参考块。也就是说,IBC可以使用本公开中描述的帧间预测技术中的至少一种。

[0088] 帧内预测器332可以通过参考当前图片中的样本来预测当前块。根据预测模式,参考样本可以位于当前块的附近或与当前块分开。在帧内预测中,预测模式可以包括多种非定向模式和多种定向模式。帧内预测器332可以通过使用应用于邻近块的预测模式来确定应用于当前块的预测模式。

[0089] 帧间预测器331可以基于参照图片上的运动向量所指定的参考块(参考样本阵列)来推导针对当前块的预测块。此时,为了减少在帧间预测模式下发送的运动信息的量,可以基于邻近块与当前块之间的运动信息的相关性以块、子块或样本为基础来预测运动信息。运动信息可以包括运动向量和参照图片索引。运动信息还可以包括帧间预测方向(L0预测、L1预测、Bi预测等)信息。在帧间预测的情况下,邻近块可以包括当前图片中存在的空间邻近块和参照图片中存在的时间邻近块。例如,帧间预测器331可以基于邻近块来配置运动信息候选列表,并且基于接收到的候选选择信息来推导当前块的运动向量和/或参照图片索引。可以基于各种预测模式来执行帧间预测,并且关于预测的信息可以包括指示针对当前块的帧间预测的模式的信息。

[0090] 加法器340可以通过将所获得的残差信号与从预测器330输出的预测信号(预测块、预测样本阵列)相加来生成重构信号(重构图片、重构块、重构样本阵列)。当如在应用跳变模式的情况下一样没有针对处理目标块的残差时,可以将预测块用作重构块。

[0091] 加法器340可以被称为重构器或重构块生成器。所生成的重构信号可以被用于当前块中的下一处理目标块的帧内预测,并且如随后描述的,可以通过滤波来输出或者被用于下一图片的帧间预测。

[0092] 此外,在图片解码处理中,可以应用具有色度缩放的亮度映射(LMCS)。

[0093] 滤波器350可以通过向重构后的信号应用滤波来改善主观/客观视频质量。例如,滤波器350可以通过向重构图片应用各种滤波方法来生成修改后的重构图片,并且可以将修改后的重构图片发送到存储器360中,尤其是发送到存储器360的DPB中。各种滤波方法可以包括例如解块滤波、样本自适应偏移、自适应环形滤波器、双边滤波器等。

[0094] 已存储在存储器360的DPB中的(修改后的)重构图片可以被用作帧间预测器331中的参照图片。存储器360可以存储从中已推导(或解码了)运动信息的当前图片中的块的运动信息和/或已经重构的图片中的块的运动信息。所存储的运动信息可以被发送到帧间预测器331,以被用作邻近块的运动信息或时间邻近块的运动信息。存储器360可以存储当前图片中的重构块的重构样本,并且将它们发送到帧内预测器332。

[0095] 在本说明书中,在解码设备300的预测器330、解量化器321、逆变换器322和滤波器

350中描述的示例可以分别类似地或对应地应用于编码设备200的预测器220、解量化器234、逆变换器235和滤波器260。

[0096] 如上所述,执行预测,以便提高执行视频编码时的压缩效率。据此,可以生成包括针对作为编码目标块的当前块的预测样本的预测块。这里,预测块包括空间域(或像素域)中的预测样本。可以在编码设备和解码设备中相同地推导预测块,并且编码设备可以通过向解码设备发信号通知并非原始块本身的原始样本值而是关于原始块与预测块之间的残差的信息(残差信息)来提高图像编码效率。解码设备可以基于残差信息来推导包括残差样本的残差块,通过将残差块与预测块相加来生成包括重构样本的重构块,并且生成包括重构块的重构图片。

[0097] 可以通过变换过程和量化过程来生成残差信息。例如,编码设备可以推导原始块与预测块之间的残差块,通过对残差块中所包括的残差样本(残差样本阵列)执行变换过程来推导变换系数,并且通过对变换系数执行量化过程来推导量化变换系数,使得它可以(通过比特流)向解码设备发信号通知关联的残差信息。这里,残差信息可以包括量化变换系数的值信息、位置信息、变换技术、变换核、量化参数等。解码设备可以基于残差信息来执行量化/解量化过程并且推导残差样本(或残差样本块)。解码设备可以基于预测块和残差块来生成重构块。编码设备可以通过对量化变换系数进行解量化/逆变换来推导残差块以便作为下一个图片的帧间预测的参考,并且可以基于此来生成重构图片。

[0098] 图4例示了应用本公开的内容流系统的结构。

[0099] 此外,应用本公开的内容流系统可以大致包括编码服务器、流服务器、web服务器、媒体存储装置、用户设备和多媒体输入装置。

[0100] 编码服务器用于将从诸如智能电话、相机、摄像机等的多媒体输入装置输入的内容压缩成数字数据以生成比特流,并且向流服务器发送它。作为另一示例,在诸如智能电话、相机、摄像机等的多媒体输入装置直接生成比特流的情况下,可以省略编码服务器。可以通过应用本公开的编码方法或比特流生成方法来生成比特流。并且流服务器可以在发送或接收比特流的过程期间临时存储比特流。

[0101] 流服务器基于用户的请求通过web服务器向用户装置发送多媒体数据,web服务器用作向用户通知存在什么服务的器具。当用户请求用户想要的服务时,web服务器将请求传送至流服务器,并且流服务器向用户发送多媒体数据。在此而言,内容流系统可以包括单独的控制服务器,并且在这种情况下,控制服务器用于控制内容流系统中的相应装置之间的命令/响应。

[0102] 流服务器可以从媒体存储装置和/或编码服务器接收内容。例如,在从编码服务器接收内容的情况下,可以实时地接收内容。在这种情况下,为了平稳地提供流服务,流服务器可以存储比特流达预定时间。

[0103] 例如,用户装置可以包括移动电话、智能电话、膝上型计算机、数字广播终端、个人数字助理(PDA)、便携式多媒体播放器(PMP)、导航仪、板式PC、平板PC、超级本、可穿戴装置(例如,手表型终端(智能手表)、眼镜型终端(智能眼镜)、头戴式显示器(HMD))、数字TV、台式计算机、数字标牌等。内容流系统中的各个服务器可以作为分布式服务器操作,并且在这种情况下,由各个服务器接收到的数据可以用分布式方式进行处理。

[0104] 图5示意性地例示了根据本公开的实施方式的多变换技术。

[0105] 参照图5,变换器可以对应于前述图2的编码设备中的变换器,并且逆变换器可以对应于前述图2的编码设备中的逆变换器、或者图3的解码设备中的逆变换器。

[0106] 变换器可以通过基于残差块中的残差样本(残差样本阵列)执行一次变换来推导(一次)变换系数(S510)。该一次变换可以被称为核心变换。在本文中,一次变换可以基于多变换选择(MTS),并且当将多变换用作一次变换时,可以将其称为多核心变换。

[0107] 多核心变换可以表示附加地使用离散余弦变换(DCT)类型2和离散正弦变换(DST)类型7、DCT类型8和/或DST类型1进行变换的方法。也就是说,多核心变换可以表示基于从DCT类型2、DST类型7、DCT类型8和DST类型1当中选择的多个变换核将空间域的残差信号(或残差块)变换为频域的变换系数(或一次变换系数)的变换方法。在本文中,从变换器的视角来看一次变换系数可以称为临时变换系数。

[0108] 换句话说,当应用常规变换方法时,可以通过基于DCT类型2对残差信号(或残差块)应用从空间域到频率域的变换来生成变换系数。与此不同,当应用多核心变换时,可以通过基于DCT类型2、DST类型7、DCT类型8和/或DST类型1对残差信号(或残差块)应用从空间域到频率域的变换来生成变换系数(或一次变换系数)。在本文中,DCT类型2、DST类型7、DCT类型8和DST类型1可以被称为变换类型、变换核或变换核心。可以基于基函数来定义这些DCT/DST变换类型。

[0109] 当执行多核心变换时,可以从变换核当中选择用于目标块的垂直变换核和水平变换核,可以基于垂直变换核对目标块执行垂直变换,并且可以基于水平变换核对目标块执行水平变换。这里,水平变换可以指示对目标块的水平分量的变换,并且垂直变换可以指示对目标块的垂直分量的变换。可以基于包括残差块的目标(CU或子块)的预测模式和/或变换索引来适应性地确定垂直变换核/水平变换核。

[0110] 此外,根据示例,如果通过应用MTS来执行一次变换,则可以通过将特定基函数设置为预定值并组合要在垂直变换或水平变换中应用的基函数来设置变换核的映射关系。例如,当水平变换核表示为trTypeHor,并且垂直方向变换核表示为trTypeVer时,可以将值为0的trTypeHor或trTypeVer设置给DCT2,将值为1的trTypeHor或trTypeVer设置给DST7,并且将值为2的trTypeHor或trTypeVer可以设置给DCT8。

[0111] 在这种情况下,MTS索引信息可以被编码并且发信号通知到解码设备以指示多个变换核集中的任何一个。例如,MTS索引0可以指示trTypeHor和trTypeVer值均为0,MTS索引1可以指示trTypeHor和trTypeVer值均为1,MTS索引2可以指示trTypeHor值为2并且trTypeVer值为1,MTS索引为3可以指示trTypeHor值为1并且trTypeVer值为2,而MTS索引4可以指示trTypeHor和trTypeVer值均为2。

[0112] 在一个示例中,根据MTS索引信息的变换核集在下表中示出。

[0113] [表1]

[0114]

tu_mts_idx[x0][y0]	0	1	2	3	4
trTypeHor	0	1	2	1	2
trTvpeVer	0	1	1	2	2

[0115] 变换器可以基于(一次)变换系数执行二次变换以推导修改的(二次)变换系数(S520)。一次变换是从空间域到频域的变换,而二次变换是指使用(一次)变换系数之间存在的相关性,变换为更紧凑的表示。二次变换可以包括不可分离的变换。在这种情况下,二

次变换可以称为不可分离的二次变换(NSST)或模式相关的不可分离的二次变换(MDNSST)。NSST可以表示基于不可分离的变换矩阵对通过一次变换推导出的(一次)变换系数进行二次变换以生成针对残差信号的修改的变换系数(或二次变换系数)的变换。这里,基于不可分离的变换矩阵,可以将变换一次应用于(一次)变换系数而无需将垂直变换和水平变换分离(或独立地应用水平/垂直变换)。换句话说,NSST不单独应用于垂直方向和水平方向上的(一次)变换系数,并且可以表示例如将二维信号(变换系数)通过特定的预定方向(例如,行优先方向或列优先方向)重新布置为一维信号并然后基于不可分离的变换矩阵生成修改的变换系数(或二次变换系数)的变换方法。例如,行优先顺序是针对M×N个块以第一行、第二行、...和第N行的顺序设置成排,而列优先顺序是针对M×N个块以第一列、第二列、...和第M列的顺序设置成排。NSST可以应用于配置有(一次)变换系数的块(在下文中,称为变换系数块)的左上区域。例如,当变换系数块的宽度W和高度H二者都为8或更大时,可以将8×8 NSST应用于变换系数块的左上8×8区域。此外,在变换系数块的宽度(W)和高度(H)二者都为4或更大的同时,当变换系数块的宽度(W)或高度(H)小于8时,4×4 NSST可以应用于变换系数块的左上min(8,W)×min(8,H)区域。然而,实施方式不限于此,例如,即使仅满足变换系数块的宽度W或高度H为4或更大的条件,也可以将4×4 NSST应用于变换系数块的左上端min(8,W)×min(8,H)区域。

[0116] 具体地,例如,如果使用4×4输入块,则不可分离的二次变换可以如下执行。

[0117] 4×4输入块X可以表示如下。

[0118] [式1]

[0119] $$X=\begin{bmatrix} X_{00} & X_{01} & X_{02} & X_{03} \\ X_{10} & X_{11} & X_{12} & X_{13} \\ X_{20} & X_{21} & X_{22} & X_{23} \\ X_{30} & X_{31} & X_{32} & X_{33} \end{bmatrix}$$

[0120] 如果X以向量的形式表示,则向量$\vec{X}$可以如下表示。

[0121] [式2]

[0122] $$\vec{X}=[X_{00}\ \ X_{01}\ \ X_{02}\ \ X_{03}\ \ X_{10}\ \ X_{11}\ \ X_{12}\ \ X_{13}\ \ X_{20}\ \ X_{21}\ \ X_{22}\ \ X_{23}\ \ X_{30}\ \ X_{31}\ \ X_{32}\ \ X_{33}]^T$$

[0123] 在式2中,向量$\vec{X}$是通过根据行优先顺序重新布置式1的二维块X而获得的一维向量。

[0124] 在这种情况下,可以如下计算不可分离的二次变换。

[0125] [式3]

[0126] $$\vec{F}=T\cdot\vec{X}$$

[0127] 在此式中,$\vec{F}$表示变换系数向量,而T表示16×16(不可分离的)变换矩阵。

[0128] 通过前述式3,可以推导16×1变换系数向量$\vec{F}$,并且可以通过扫描顺序(水平、垂直和对角线等)将向量$\vec{F}$重新组织为4×4块。然而,上述计算是示例,并且超立方体-吉文斯变换(HyGT)等也可以用于不可分离的二次变换的计算,以便降低不可分离的二次变换的计算复杂度。

[0129] 此外,在不可分离的二次变换中,可以将变换核(或变换核心、变换类型)选择为模式相关。在这种情况下,模式可以包括帧内预测模式和/或帧间预测模式。

[0130] 如上所述,可以基于以变换系数块的宽度(W)和高度(H)为基础确定的8×8变换或4×4变换来执行不可分离的二次变换。8×8变换是指当W和H二者都等于或大于8时可应用于变换系数块中包含的8×8区域的变换,并且8×8区域可以是变换系数块中的左上8×8区域。类似地,4×4变换是指当W和H二者都等于或大于4时可应用于变换系数块中包含的4×4区域的变换,并且4×4区域可以是变换系数块中的左上4×4区域。例如,8×8变换核矩阵可以是64×64/16×64矩阵,而4×4变换核矩阵可以是16×16/8×16矩阵。

[0131] 这里,为了选择模式相关的变换核,可以针对8×8变换和4×4变换二者配置用于不可分离的二次变换的每个变换集二个不可分离的二次变换核,并且可以存在四个变换集。也就是说,可以针对8×8变换配置四个变换集,并且可以针对4×4变换配置四个变换集。在这种情况下,针对8×8变换的四个变换集中的每个变换集可以包括二个8×8变换核,并且针对4×4变换的四个变换集中的每个变换集可以包括二个4×4变换核。

[0132] 然而,随着变换的大小(即,变换所应用于的区域的大小)可以为例如除了8×8或4×4之外的大小,集的数量可以是n,并且每个集中的变换核的数量可以是k。

[0133] 变换集可以被称为NSST集或LFNST集。可以例如基于当前块(CU或子块)的帧内预测模式来选择变换集当中的特定集。低频不可分离的变换(LFNST)可以是缩减不可分离的变换的示例,其将稍后描述,并且表示用于低频分量的不可分离的变换。

[0134] 作为参考,例如,帧内预测模式可以包括两个非定向(或非角度)帧内预测模式和65个定向(或角度)帧内预测模式。非定向帧内预测模式可以包括0号的平面帧内预测模式和1号的DC帧内预测模式,并且定向帧内预测模式可以包括2号至66号的65个帧内预测模式。然而,这是示例,并且即使帧内预测模式的数量不同也可以应用本文档。此外,在一些情况下,还可以使用67号帧内预测模式,并且67号帧内预测模式可以表示线性模型(LM)模式。

[0135] 图6示意性地示出了65个预测方向的帧内定向模式。

[0136] 参照图6,基于具有左上对角预测方向的帧内预测模式34,帧内预测模式可以划分为具有水平方向性的帧内预测模式和具有垂直方向性的帧内预测模式。在图6中,H和V分别标示水平方向性和垂直方向性,并且数字-32至32指示样本网格位置上的1/32单位的位移。这些数字可以表示对于模式索引值的偏移。帧内预测模式2至33具有水平方向性,并且帧内预测模式34至66具有垂直方向性。严格地说,帧内预测模式34可以被视为既不是水平的也不是垂直的,但在确定二次变换的变换集时可以被分类为属于水平方向性。这是因为输入数据被转置以用于基于帧内预测模式34对称的垂直定向模式,并且针对水平模式的输入数据对准方法用于帧内预测模式34。对输入数据进行转置意指将二维的M×N块数据的行和列切换成N×M数据。帧内预测模式18和帧内预测模式50可以分别表示水平帧内预测模式和垂直帧内预测模式,并且帧内预测模式2可以被称为右上对角帧内预测模式,因为帧内预测模式2具有左参考像素并且在右上方向上执行预测。类似地,帧内预测模式34可以被称为右下对角帧内预测模式,而帧内预测模式66可以被称为左下对角帧内预测模式。

[0137] 根据示例,可以映射根据帧内预测模式的四个变换集,例如,如下表所示。

[0138] [表2]

[0139]

lfnstPredModeIntra	lfnstTrSetIdx

lfnstPredModeIntra＜0	1
0<＝lfnstPredModeIntra<＝1	0
2<＝lfnstPredModeIntra<＝12	1
13＜＝lfnstPredModeIntra＜＝23	2
24<＝lfnstPredModeIntra<＝44	3
45<＝lfnstPredModeIntra<＝55	2
56<＝lfnstPredModeIntra<＝80	1
81<＝lfnstPredModeIntra<＝83	0

[0140] 如表2所示,根据帧内预测模式,四个变换集中的任何一个,即,lfnstTrSetIdx,可以映射到四个索引(即,0至3)中的任何一个。

[0141] 当确定特定集用于不可分离的变换时,可以通过不可分离的二次变换索引来选择特定集中的k个变换核之一。编码设备可以基于率失真(RD)校验来推导指示特定变换核的不可分离的二次变换索引,并且可以将不可分离的二次变换索引发信号通知给解码设备。解码设备可以基于不可分离的二次变换索引来选择特定集中的k个变换核中的一个。例如,lfnst索引值0可以指代第一不可分离的二次变换核,lfnst索引值1可以指代第二不可分离的二次变换核,lfnst索引值2可以指代第三不可分离的二次变换核。另选地,lfnst索引值0可以指示第一不可分离的二次变换没有被应用于目标块,并且lfnst索引值1至3可以指示三个变换核。

[0142] 变换器可以基于所选择的变换核来执行不可分离的二次变换,并且可以获得修改的(二次)变换系数。如上所述,修改的变换系数可以被推导为通过量化器量化的变换系数,并且可以被编码并发信号通知给解码设备,并且被传送到编码设备中的解量化器/逆变换器。

[0143] 此外,如上所述,如果省略了二次变换,则可以将作为一次(可分离的)变换的输出的(一次)变换系数推导为如上所述通过量化器量化的变换系数,并且可以被编码并发信号通知给解码设备,并传送到编码设备中的解量化器/逆变换器。

[0144] 逆变换器可以与在上述变换器中已经执行的顺序相反的顺序执行一系列过程。逆变换器可以接收(解量化的)变换系数,并且通过执行二次(逆)变换来推导(一次)变换系数(S550),并且可以通过对(一次)变换系数执行一次(逆)变换来获得残差块(残差样本)(S560)。就此而言,从逆变换器的视角来看,一次变换系数可以被称为修改的变换系数。如上所述,编码设备和解码设备可以基于残差块和预测块来生成重构块,并且可以基于重构块来生成重构图片。

[0145] 解码设备还可以包括二次逆变换应用确定器(或用于确定是否应用二次逆变换的元件)和二次逆变换确定器(或用于确定二次逆变换的元件)。二次逆变换应用确定器可以确定是否应用二次逆变换。例如,二次逆变换可以是NSST、RST或LFNST,并且二次逆变换应用确定器可以基于通过解析比特流而获得的二次变换标志来确定是否应用二次逆变换。在另一示例中,二次逆变换应用确定器可以基于残差块的变换系数来确定是否应用二次逆变换。

[0146] 二次逆变换确定器可以确定二次逆变换。在这种情况下,二次逆变换确定器可以基于根据帧内预测模式指定的LFNST(NSST或RST)变换集来确定应用于当前块的二次逆变

换。在实施方式中,可以取决于一次变换确定方法来确定二次变换确定方法。可以根据帧内预测模式来确定一次变换和二次变换的各种组合。此外,在示例中,二次逆变换确定器可以基于当前块的大小来确定应用二次逆变换的区域。

[0147] 此外,如上所述,如果省略二次(逆)变换,则可以接收(解量化的)变换系数,可以执行一次(可分离的)逆变换,并且可以获得残差块(残差样本)。如上所述,编码设备和解码设备可以基于残差块和预测块来生成重构块,并且可以基于重构块来生成重构图片。

[0148] 此外,在本公开中,可以在NSST的概念中应用其中减小了变换矩阵(核)的大小的缩减二次变换(RST),以便减少不可分离的二次变换所需的计算量和存储量。

[0149] 此外,本公开中描述的变换核、变换矩阵以及构成变换核矩阵的系数,即,核系数或矩阵系数,可以以8比特来表示。这可以是在解码设备和编码设备中实现的条件,并且与现有的9比特或10比特相比,可以减少存储变换核所需的存储量,并且可以合理地适应性能劣化。另外,以8比特表示核矩阵可以允许使用小的乘法器,并且可以更适合于用于最佳软件实现的单指令多数据(SIMD)指令。

[0150] 在本说明书中,术语“RST”可以是指基于大小根据缩减因子而减小的变换矩阵来对目标块的残差样本执行的变换。在执行缩减变换的情况下,由于变换矩阵的大小的减小,可以减少变换所需的计算量。也就是说,RST可以用于解决在大小大的块的变换或不可分离的变换时发生的计算复杂性问题。

[0151] RST可以被称为诸如缩减变换、缩减二次变换、缩小变换、简化变换和简单变换等之类的各种术语,并且RST可以被称为的名称不限于所列示例。另选地,由于RST主要在变换块中的包括非零系数的低频区域中执行,因此它可以被称为低频不可分离的变换(LFNST)。变换索引可以被称作LFNST索引。

[0152] 此外,当基于RST执行二次逆变换时,编码设备200的逆变换器235和解码设备300的逆变换器322可以包括:逆缩减二次变换器,其基于变换系数的逆RST来推导修改的变换系数;以及逆一次变换器,其基于修改的变换系数的逆一次变换来推导目标块的残差样本。逆一次变换是指应用于残差的一次变换的逆变换。在本公开中,基于变换推导变换系数可以是指通过应用变换来推导变换系数。

[0153] 图7是例示根据本公开的实施方式的RST的图。

[0154] 在本公开中,“目标块”可以指代要编码的当前块,残差块或变换块。

[0155] 在根据示例的RST中,可以将N维向量映射到位于另一个空间中的R维向量,从而可以确定缩减变换矩阵,其中R小于N。N可以是指应用了变换的块的侧边的长度的平方,或与应用了变换的块相对应的变换系数的总数,并且缩减因子可以是指R/N值。缩减因子可以被称为缩减因子、缩小因子、简化因子、简单因子或其它各种术语。此外,R可以被称为缩减系数,但是根据情况,缩减因子可以是指R。此外,根据情况,缩减因子可以是指N/R值。

[0156] 在示例中,可以通过比特流来发信号通知缩减因子或缩减系数,但是示例不限于此。例如,可以在编码设备200和解码设备300中的每一个中存储针对缩减因子或缩减系数的预定值,并且在这种情况下,可以不单独发信号通知缩减因子或缩减系数。

[0157] 根据示例的缩减变换矩阵的大小可以是小于N×N(常规变换矩阵的大小)的R×N,并且可以如下面的式4所限定。

[0158] [式4]

[0159] $$T_{RxN}=\begin{bmatrix} t_{11} & t_{12} & t_{13} & \cdots & t_{1N} \\ t_{21} & t_{22} & t_{23} & & t_{2N} \\ & \vdots & & \ddots & \vdots \\ t_{R1} & t_{R2} & t_{R3} & \cdots & t_{RN} \end{bmatrix}$$

[0160] 图7的(a)中所示的缩减变换块中的矩阵T可以是指式4的矩阵$T_{R\times N}$。如图7的(a)所示,当将缩减变换矩阵$T_{R\times N}$乘以目标块的残差样本时,可以推导当前块的变换系数。

[0161] 在示例中,如果应用了变换的块的大小是8×8并且R=16(即,R/N=16/64=1/4),则根据图7的(a)的RST可以被表示为以下式5所示的矩阵运算。在这种情况下,存储和乘法计算可以通过缩减因子缩减至大约1/4。

[0162] 在本公开中,矩阵运算可以理解为通过将列向量与设置在列向量的左侧的矩阵相乘来获得列向量的运算。

[0163] [式5]

[0164] $$\begin{bmatrix} t_{1,1} & t_{1,2} & t_{1,3} & \cdots & t_{1,64} \\ t_{2,1} & t_{2,2} & t_{2,3} & & t_{2,64} \\ & \vdots & & \ddots & \vdots \\ t_{16,1} & t_{16,2} & t_{16,3} & \cdots & t_{16,64} \end{bmatrix} \times \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ \vdots \\ r_{64} \end{bmatrix}$$

[0165] 在式5中,r_1至r_{64}可以表示目标块的残差样本,并且具体地可以是通过应用一次变换而生成的变换系数。作为式5的计算的结果,可以推导目标块的变换系数c_i,并且推导c_i的过程可以如式6所示。

[0166] [式6]

[0167]
```
针对i从1到R:
    c_i = 0
    针对j从1到N:
        c_i += t_{i,j} * r_j
```

[0168] 作为式6的计算的结果,可以推导目标块的变换系数c_1至c_R。也就是说,当R=16时,可以推导目标块的变换系数c_1至c_{16}。如果应用常规变换而不是RST,并将64×64(N×N)大小的变换矩阵与64×1(N×1)大小的残差样本相乘,则因为应用了RST而针对目标块仅推导16(R)个变换系数,尽管针对目标块推导了64(N)个变换系数。由于用于目标块的变换系数的总数从N缩减到R,所以编码设备200向解码设备300发送的数据量减少,因此编码设备200与解码设备300之间的传输效率可以提高。

[0169] 当从变换矩阵的大小的视角考虑时,常规变换矩阵的大小为64×64(N×N),但缩减变换矩阵的大小缩减为16×64(R×N),因此与执行常规变换的情况相比,执行RST的情况下的存储使用率可以减小R/N比率。另外,当与使用常规变换矩阵的情况下的乘法计算的数量N×N相比时,使用缩减变换矩阵可以将乘法计算的数量(R×N)减小R/N比率。

[0170] 在示例中,编码设备200的变换器232可以通过对目标块的残差样本执行一次变换和基于RST的二次变换来推导目标块的变换系数。这些变换系数可以被传送到解码设备300的逆变换器,并且解码设备300的逆变换器322可以基于针对变换系数的逆缩减二次变换

(RST)来推导修改的变换系数,并且可以基于针对修改的变换系数的逆一次变换来推导目标块的残差样本。

[0171] 根据示例的逆RST矩阵$T_{N\times R}$的大小为比常规逆变换矩阵N×N的大小小的N×R,并且与式4中所示的缩减变换矩阵$T_{R\times N}$具有转置关系。

[0172] 图7的(b)所示的缩减逆变换块中的矩阵T^t可以是指逆RST矩阵$T_{N\times R}{}^T$(上标T是指转置)。如图7的(b)所示,当将逆RST矩阵$T_{N\times R}{}^T$乘以目标块的变换系数时,可以推导目标块的修改的变换系数或目标块的残差样本。逆RST矩阵$T_{R\times N}{}^T$可以表示为$(T_{R\times N})^T{}_{N\times R}$。

[0173] 更具体地,当逆RST被用作二次逆变换时,当逆RST矩阵$T_{N\times R}{}^T$被乘以目标块的变换系数时,可以推导目标块的修改的变换系数。此外,可以将逆RST用作逆一次变换,并且在这种情况下,当将逆RST矩阵$T_{N\times R}{}^T$与目标块的变换系数相乘时,可以推导目标块的残差样本。

[0174] 在示例中,如果应用逆变换的块的大小是8×8并且R=16(即,R/N=16/64=1/4),则根据图7的(b)的RST可以被表示为以下式7所示的矩阵运算。

[0175] [式7]

[0176]
$$\begin{bmatrix} t_{1,1} & t_{2,1} & \cdots & t_{16,1} \\ t_{1,2} & t_{2,2} & \cdots & t_{16,1} \\ t_{1,2} & t_{2,3} & & t_{16,1} \\ \vdots & \vdots & & \vdots \\ \vdots & & \ddots & \vdots \\ t_{1,64} & t_{2,64} & \cdots & t_{16,64} \end{bmatrix} \times \begin{bmatrix} c_1 \\ c_{11} \\ \vdots \\ c_{16} \end{bmatrix}$$

[0177] 在式7中,c_1至c_{16}可以表示目标块的变换系数。作为式7的计算的结果,可以推导表示目标块的修改的变换系数或目标块的残差样本的r_i,并且推导r_i的过程可以如式8所示。

[0178] [式8]

[0179]
```
针对i从1到N
    ri=0
    针对j从1到R
        ri += tji * cj
```

[0180] 作为式8的计算的结果,可以推导表示目标块的修改的变换系数或目标块的残差样本的r_1至r_N。从逆变换矩阵的大小的视角考虑,常规逆变换矩阵的大小为64×64(N×N),但逆缩减变换矩阵的大小缩减为64×16(R×N),因此与执行常规逆变换的情况相比,执行逆RST的情况下的存储使用率可以减小R/N比率。另外,当与使用常规逆变换矩阵的情况下的乘法计算的数量N×N相比时,使用逆缩减变换矩阵可以将乘法计算的数量(N×R)减少R/N比率。

[0181] 表2所示的变换集配置也可以应用于8×8RST。也就是说,可以根据表2中的变换集来应用8×8RST。由于根据帧内预测模式,一个变换集包括两个或三个变换(核),因此可以将其配置为选择包括在不应用二次变换的情况下在内的至多四个变换中的一个。在不应用二次变换的变换中,可以考虑应用恒等矩阵(identity matrix)。假设分别将索引0、1、2和3分配给四个变换(例如,可以将索引0分配给应用恒等矩阵的情况,即,不应用二次变换的情况),可以针对每个变换系数块发信号通知作为语法元素的变换索引或lfnst索引,由此指

定要应用的变换。也就是说,针对左上8×8块,通过变换索引,可以指定RST配置中的8×8NSST,或者当应用LFNST时可以指定8×8lfnst。8×8lfnst和8×8RST指代当要变换的目标块的W和H均等于或大于8时可应用于变换系数块中包括的8×8区域的变换,并且8×8区域可以是变换系数块中的左上8×8区域。类似地,4×4lfnst和4×4RST指代当目标块的W和H均等于或大于4时可应用于变换系数块中包括的4×4区域的变换,并且4×4区域可以是变换系数块中的左上4×4区域。

[0182] 根据本公开的实施方式,对于编码过程中的变换,可以仅选择48条数据,并且可以向其应用最大16×48变换核矩阵,而不是将16×64变换核矩阵应用于形成8×8区域的64条数据。此处,“最大”意味着m在m×48变换核矩阵中具有最大值16以用于生成m个系数。也就是说,当通过将m×48变换核矩阵(m≤16)应用于8×8区域来执行RST时,输入48条数据,并且生成m个系数。当m是16时,输入48条数据并且生成16个系数。也就是说,假设48条数据形成48×1向量,16×48矩阵和48×1向量依次相乘,由此生成16×1向量。这里,形成8×8区域的48条数据可以被适当地布置,由此形成48×1向量。例如,可以基于构成8×8区域当中的除了右下4×4区域之外的区域的48条数据来构造48×1向量。这里,当通过应用最大16×48变换核矩阵来执行矩阵运算时,生成16个修改的变换系数,并且可以根据扫描顺序将16个修改的变换系数布置在左上4×4区域中,并且可以用零填充右上4×4区域和左下4×4区域。

[0183] 对于解码过程中的逆变换,可以使用前述变换核矩阵的转置矩阵。也就是说,当在由解码设备执行的逆变换过程中执行逆RST或LFNST时,根据预定布置顺序在一维向量中配置应用逆RST的输入系数数据,并且可以根据预定布置顺序将通过将一维向量与在一维向量左侧的对应的逆RST矩阵相乘而获得的修改的系数向量布置到二维块中。

[0184] 总之,在变换过程中,当RST或LFNST被应用于8×8区域时,在8×8区域的除了右下区域之外的左上区域、右上区域和左下区域中的48个变换系数与16×48变换核矩阵的矩阵运算。对于矩阵运算,以一维阵列输入48个变换系数。当执行矩阵运算时,推导出16个修改的变换系数,并且可以将修改的变换系数布置于8×8区域的左上区域中。

[0185] 相反,在逆变换过程中,当将逆RST或LFNST应用于8×8区域时,可以根据扫描顺序以一维阵列输入8×8区域中的变换系数当中的对应于8×8区域的左上区域的16个变换系数,并且可以经历与48×16变换核矩阵的矩阵运算。也就是说,矩阵运算可以表示为(48×16矩阵)*(16×1变换系数向量)=(48×1修改的变换系数向量)。这里,n×1向量可以被解释为具有与n×1矩阵相同的含义,并且因此可以被表示为n×1列向量。此外,*表示矩阵乘法。当执行矩阵运算时,可以推导出48个修改的变换系数,并且可以将48个修改的变换系数布置在8×8区域中的除了右下区域之外的左上区域、右上区域和左下区域中。

[0186] 当二次逆变换基于RST时,编码设备200的逆变换器235和解码设备300的逆变换器322可以包括用于基于对变换系数的逆RST推导修改的变换系数的逆缩减二次变换器以及用于基于对修改的变换系数的逆一级变换来推导目标块的残差样本的逆一次变换器。逆一次变换指代应用于残差的一次变换的逆变换。在本公开中,基于变换推导变换系数可以指代通过应用变换来推导变换系数。

[0187] 上面描述的非分离变换(LFNST)将如下详细描述。LFNST可以包括由编码设备进行的正向变换和由解码设备进行的逆变换。

[0188] 编码设备接收在应用一次(核心)变换之后推导出的结果(或结果的一部分)作为输入,并且应用正向二次变换(二次变换)。

[0189] [式9]

[0190] $y=G^{T}x$

[0191] 在式9中,x和y分别是二次变换的输入和输出,G是表示二次变换的矩阵,并且变换基向量由列向量组成。在逆LFNST的情况下,当变换矩阵G的维度表示为[行的数量×列的数量]时,在正向LFNST的情况下,矩阵G的转置变成G^{T}的维度。

[0192] 对于逆LFNST,矩阵G的维度是[48×16]、[48×8]、[16×16]、[16×8],并且[48×8]矩阵和[16×8]矩阵是分别从[48×16]矩阵和[16×16]矩阵的左侧采样的8个变换基向量的部分矩阵。

[0193] 另一方面,对于正向LFNST,矩阵G^{T}的维度是[16×48]、[8×48]、[16×16]、[8×16],并且[8×48]矩阵和[8×16]矩阵是通过分别从[16×48]矩阵和[16×16]矩阵的上部采样8个变换基向量而获得的部分矩阵。

[0194] 因此,在正向LFNST的情况下,[48×1]向量或[16×1]向量可以作为输入x,并且[16×1]向量或[8×1]向量可以作为输出y。在视频编码和解码中,正向一次变换的输出是二维(2D)数据,因此为了构造[48×1]向量或[16×1]向量作为输入x,需要通过将作为正向变换的输出的2D数据适当地布置来构造一维向量。

[0195] 图8是例示根据示例的将正向一次变换的输出数据布置成一维向量的顺序的图。图8的(a)和(b)的左图示出了用于构造[48×1]向量的顺序,并且图8的(a)和(b)的右图示出了用于构造[16×1]向量的顺序。在LFNST的情况下,可以通过将2D数据按与图8的(a)和(b)中相同的次序顺序地布置来获得一维向量x。

[0196] 可以根据当前块的帧内预测模式确定正向一次变换的输出数据的布置方向。例如,在当前块的帧内预测模式相对于对角线方向处于水平方向时,可以按图8的(a)的顺序布置正向一次变换的输出数据,并且在当前块的帧内预测模式相对于对角线方向处于垂直方向时,可以按图8的(b)的顺序布置正向一次变换的输出数据。

[0197] 根据示例,可以应用不同于图8的(a)和(b)的布置顺序的布置顺序,并且为了推导与应用图8的(a)和(b)的布置顺序时相同的结果(y向量),可以根据布置顺序重新布置矩阵G的列向量。也就是说,可以重新布置G的列向量,使得构成x向量的每个元素总是乘以相同的变换基向量。

[0198] 由于通过式9推导的输出y是一维向量,因此当在使用正向二次变换的结果作为输入的过程中(例如,在执行量化或残差编码的过程中)需要二维数据作为输入数据时,式9的输出y向量需要再次被适当地布置为2D数据。

[0199] 图9是例示根据示例的将正向二次变换的输出数据布置成二维块的顺序的图。

[0200] 在LFNST的情况下,输出值可以根据预定扫描顺序布置在2D块中。图9的(a)示出了当输出y是[16×1]向量时,根据对角线扫描顺序将输出值布置在2D块的16个位置处。图9的(b)示出了当输出y是[8×1]向量时,根据对角线扫描顺序将输出值布置在2D块的8个位置处,并且用零填充其余的8个位置。图9的(b)中的X指示它被填充有零。

[0201] 根据另一示例,由于可以预设在执行量化或残差编码时处理输出向量y的顺序,因此输出向量y可以不布置在如图9中所示的2D块中。然而,在残差编码的情况下,可以在2D块

(例如,4×4)单元(例如,CG(系数组))中执行数据编码,并且在此情况下,根据如图9的对角线扫描顺序中的特定顺序来布置数据。

[0202] 此外,解码设备可以通过根据用于逆变换的预设扫描顺序来布置通过解量化过程输出的二维数据来配置一维输入向量y。输入向量y可以通过下式输出为输出向量x。

[0203] [式10]

[0204] $x=Gy$

[0205] 在逆LFNST的情况下,可以通过将作为[16×1]向量或[8×1]向量的输入向量y乘以G矩阵来推导输出向量x。对于逆LFNST,输出向量x可以是[48×1]向量或[16×1]向量。

[0206] 输出向量x根据图8中所示的顺序布置在二维块中,并且被布置为二维数据,并且该二维数据成为逆一次变换的输入数据(或输入数据的一部分)。

[0207] 因此,逆二次变换整体上是正向二次变换过程的相反,并且在逆变换的情况下,与在正向方向上不同,首先应用逆二次变换,然后应用逆一次变换。

[0208] 在逆LFNST中,可以选择8个[48×16]矩阵和8个[16×16]矩阵中的一个作为变换矩阵G。是应用[48×16]矩阵还是应用[16×16]矩阵取决于块的大小和形状。

[0209] 另外,可以从如上表2所示的四个变换集中推导8个矩阵,并且每个变换集可以由两个矩阵组成。根据帧内预测模式确定在4个变换集当中使用哪个变换集,并且更具体地,基于通过考虑广角帧内预测(WAIP)而扩展的帧内预测模式的值来确定变换集。通过索引信令来推导从构成所选择的变换集的两个矩阵当中选择哪个矩阵。更具体地,0、1和2可以作为发送的索引值,0可以指示不应用LFNST,并且1和2可以指示构成基于帧内预测模式值选择的变换集的两个变换矩阵中的任何一个。

[0210] 图10是例示根据本文档的实施方式的广角帧内预测模式的图。

[0211] 一般帧内预测模式值可以具有从0到66以及从81到83的值,并且由于WAIP而扩展的帧内预测模式值可以具有所示的从-14到83的值。从81到83的值指示CCLM(跨分量线性模型)模式,并且从-14到-1的值和从67到80的值指示由于WAIP应用而扩展的帧内预测模式。

[0212] 当当前预测块的宽度大于高度时,上参考像素通常更接近要预测的块内部的位置。因此,在左下方向上比在右上方向上进行预测可以更准确。相反,当块的高度大于宽度时,左参考像素通常更接近要预测的块内部的位置。因此,在右上方向上比在左下方向上进行预测可以更准确。因此,将重映射(即,模式索引修改)应用到广角帧内预测模式的索引可以是有利的。

[0213] 当应用广角帧内预测时,关于现有帧内预测的信息可以被发信号通知,并且在信息被解析之后,该信息可以被重映射到广角帧内预测模式的索引。因此,可以不改变用于特定块(例如,特定大小的非正方形块)的帧内预测模式的总数量,也就是说,帧内预测模式的总数量是67,并且可以不改变用于特定块的帧内预测模式编码。

[0214] 下表3示出了通过将帧内预测模式重映射到广角帧内预测模式来推导修改的帧内模式的过程。

[0215] [表3]

[0216]

此过程的输入为：

- 变量predModeIntra，其规定帧内预测模式，
- 变量nTbW，其规定变换块宽度，
- 变量nTbH，其规定变换块高度，
- 变量cIdx，其规定当前块的颜色分量。

此过程的输出为修改的帧内预测模式predModeIntra。

变量nW和nH被推导如下：

- 如果IntraSubPartitionsSplitType等于ISP_NO_SPLIT或者cIdx不等于0，则应用以下：

$$nW = nTbW \quad (8\text{-}97)$$

$$nH = nTbH \quad (8\text{-}98)$$

- 否则（IntraSubPartitionsSplitType不等于ISP_NO_SPLIT并且cIdx等于0），则应用以下：

$$nW = nCbW \quad (8\text{-}99)$$

$$nH = nCbH \quad (8\text{-}100)$$

变量whRatio被设置为等于Abs(Log2(nW/nH))。

针对非正方形块（nW不等于nH），帧内预测模式predModeIntra被修改如下：

- 如果以下条件全部为真，则predModeIntra被设置为等于(predModeIntra+65)。
  - nW大于nH
  - predModeIntra大于或等于2
  - predModeIntra小于(whRatio > 1) ? (8 + 2 * whRatio) : 8
- 否则，如果以下条件全部为真，则predModeIntra被设置为等于(predModeIntra-67)。
  - nH大于nW
  - predModeIntra小于或等于66
  - predModeIntra大于(whRatio > 1) ? (60 - 2 * whRatio) : 60

[0217] 在表3中,扩展的帧内预测模式值最终存储在predModeIntra变量中,并且ISP_NO_SPLIT指示CU块不通过当前在VVC标准中采用的帧内子分区(ISP)技术划分成子分区,并且0、1和2的cIdx变量值分别指示亮度分量、Cb分量和Cr分量的情况。表3所示的log2函数返回基数为2的log值,并且Abs函数返回绝对值。

[0218] 指示帧内预测模式的变量predModeIntra以及变换块的高度和宽度等用作广角帧内预测模式映射过程的输入值,并且输出值是修改的帧内预测模式predModeIntra。变换块或编码块的高度和宽度可以为用于帧内预测模式的重映射的当前块的高度和宽度。此时,反映宽度与宽度的比率的变量whRatio可以被设置为Abs(Log2(nW/nH))。

[0219] 对于非正方形块,帧内预测模式可以划分成两种情况并且被修改。

[0220] 首先,如果条件(1)至(3)全部被满足,(1)当前块的宽度大于高度、(2)在修改之前的帧内预测模式等于或大于2、以及(3)帧内预测模式当变量whRatio大于1时小于被推导为(8+2*whRatio)的值并且当变量whRatio小于或等于1时小于8(predModeIntra小于(whRatio>1)?(8+2*whRatio):8),则帧内预测模式被设置为比predModeIntra大65的值[predModeIntra被设置为等于(predModeIntra+65)]。

[0221] 如果与以上不同,即,如果条件(1)至(3)被满足,(1)当前块的高度大于宽度、(2)

在修改之前的帧内预测模式小于或等于66、以及(3)帧内预测模式当whRatio大于1时大于被推导为(60-2*whRatio)的值并且当whRatio小于或等于1时大于60(predModeIntra大于(whRatio>1)?(60-2*whRatio):60),则帧内预测模式被设置为比predModeIntra小67的值[predModeIntra被设置为等于(predModeIntra-67)]。

[0222] 上表2示出了如何在LFNST中基于由WAIP扩展的帧内预测模式值选择变换集。如图10所示,模式14到33和模式35到80关于模式34周围的预测方向对称。例如,模式14和模式54关于对应于模式34的方向是对称的。因此,相同的变换集应用于位于相互对称的方向上的模式,并且这种对称性也反映在表2中。

[0223] 此外,假设模式54的正向LFNST输入数据与模式14的正向LFNST输入数据对称。例如,对于模式14和模式54,根据图8的(a)和图8的(b)所示的布置顺序将二维数据重新布置为一维数据。另外,可以看出,图8的(a)和图8的(b)所示的顺序的图案关于由模式34指示的方向(对角线方向)是对称的。

[0224] 此外,如上所述,由变换目标块的大小和形状来确定将[48×16]矩阵和[16×16]矩阵中的哪个变换矩阵应用于LFNST。

[0225] 图11是例示LFNST被应用于的块形状的图。图11的(a)示出了4×4块,图11的(b)示出了4×8块和8×4块,图11的(c)示出了4×N块或N×4块,其中N为16或更大,图11的(d)示出了8×8块,图11的(e)示出了M×N块,其中M≥8、N≥8并且N>8或M>8。

[0226] 在图11中,具有厚边界的块指示LFNST被应用到的区域。对于图11的(a)和(b)的块,LFNST被应用于左上4×4区域,并且对于图11的(c)的块,LFNST被单独地应用于被连续地布置的两个左上4×4区域。在图11的(a)、(b)和(c)中,由于LFNST以4×4区域为单位应用,所以该LFNST在下文中将被称为“4×4LFNST”。基于G的矩阵维度,可以应用[16×16]或[16×8]矩阵。

[0227] 更具体地,[16×8]矩阵被应用到图11的(a)的4×4块(4×4TU或4×4CU),并且[16×16]矩阵被应用到图11的(b)和(c)中的块。这是为了将最坏情况的计算复杂度调整为每个样本8次乘法。

[0228] 关于图11的(d)和(e),LFNST被应用于左上8×8区域,并且该LFNST在下文中被称为“8×8LFNST”。作为对应的变换矩阵,可以应用[48×16]矩阵或[48×8]矩阵。在正向LFNST的情况下,由于[48×1]向量(式9中的X向量)作为输入数据被输入,所以不是左上8×8区域的所有样本值都被用作正向LFNST的输入值。也就是说,如可以从图8的(a)的左侧顺序或图8的(b)的左侧顺序看出的,可以基于属于在将右下4×4块原样留下的同时的其余3个4×4块的样本来构造[48×1]向量。

[0229] [48×8]矩阵可以应用于图11的(d)中的8×8块(8×8TU或8×8CU),并且[48×16]矩阵可以应用于图11的(e)中的8×8块。这也是为了将最坏情况的计算复杂度调整为每个样本8次乘法。

[0230] 取决于块形状,当对应的正向LFNST(4×4或8×8LFNST)被应用时,生成8或16个输出数据(式9中的Y向量,[8×1]或[16×1]向量)。在正向LFNST中,由于矩阵G^T的特性,输出数据的数量等于或小于输入数据的数量。

[0231] 图12是例示根据示例的正向LFNST的输出数据的布置的图,并且示出了其中根据块形状布置正向LFNST的输出数据的块。

[0232] 在图12所示的块的左上的阴影区域对应于正向LFNST的输出数据所位于的区域，用0标记的位置指示填充有值0的样本，并且其余区域表示未被正向LFNST改变的区域。在未被LFNST改变的区域中，正向一次变换的输出数据保持不变。

[0233] 如上所述，由于所应用的变换矩阵的尺寸根据块的形状而变化，因此输出数据的数量也变化。如图12，正向LFNST的输出数据可能不完全填充左上4×4块。在图12的(a)和(d)的情况下，将[16×8]矩阵和A[48×8]矩阵分别应用于由粗线指示的块或块内部的部分区域，并且生成作为正向LFNST的输出的[8×1]向量。也就是说，根据图9的(b)所示的扫描顺序，可以仅填充8个输出数据，如图12的(a)和(d)所示，并且可以在其余的8个位置中填充0。在图11的(d)的LFNST应用的块的情况下，如图12的(d)所示，与左上4×4块相邻的右上和左下的两个4×4块也被填充有值0。

[0234] 如上所述，基本上，通过发信号通知LFNST索引，规定了是否应用LFNST和要应用的变换矩阵。如图12所示，当LFNST被应用时，由于正向LFNST的输出数据的数量可以等于或小于输入数据的数量，所以出现如下填充有零值的区域。

[0235] 1)如图12的(a)所示，来自左上4×4块中的扫描顺序上的第八个位置和后面的位置的样本，即，来自第九到第十六的样本。

[0236] 2)如图12的(d)和(e)中所示，当应用[48×16]矩阵或[48×8]矩阵时，与左上4×4块相邻的两个4×4块或者扫描顺序上的第二和第三4×4块。

[0237] 因此，如果通过检查区域1)和2)存在非零数据，则确定未应用LFNST，使得可以省略对应的LFNST索引的信令。

[0238] 根据示例，例如，在VVC标准中采用的LFNST的情况下，由于在残差编码之后执行LFNST索引的信令，因此编码设备可以通过残差编码来知道在TU或CU块内的所有位置是否存在非零数据(有效系数)。因此，编码设备可以基于非零数据的存在来确定是否执行关于LFNST索引的信令，并且解码设备可以确定是否解析LFNST索引。当非零数据不存在于以上1)和2)中指定的区域中时，执行LFNST索引的信令。

[0239] 此外，对于采用的LFNST，可以应用以下简化方法。

[0240] (i)根据示例，正向LFNST的输出数据的数量可以被限于最大值16。

[0241] 在图11的(c)的情况下，4×4LFNST可以分别应用于与左上方相邻的两个4×4区域，并且在这种情况下，可以生成最大32个LFNST输出数据。当正向LFNST的输出数据的数量被限制为最大值16时，在4×N/N×4(N≥16)块(TU或CU)的情况下，4×4LFNST仅应用于左上方的一个4×4区域，LFNST可以仅应用于图11的所有块一次。通过此，可以简化图像编码的实现方式。

[0242] (ii)根据示例，清零可以附加地应用到未应用LFNST的区域。在本文档中，清零可以表示用为0的值填充属于特定区域的所有位置。也就是说，可以将清零应用于由于LFNST而未改变的区域，并且维持正向一次变换的结果。如上所述，由于LFNST被划分为4×4LFNST和8×8LFNST，所以清零可以如下划分为两种类型((ii)-(A)和(ii)-(B))。

[0243] (ii)-(A)当应用4×4LFNST时，未应用4×4LFNST的区域可以被清零。图13是例示根据示例的在应用了4×4LFNST的块中的清零的图。

[0244] 如图13所示，针对应用了4×4LFNST的块，即，针对图12的(a)、(b)和(c)中的所有块，即使未应用LFNST的整个区域也可以用零填充。

[0245] 另一方面,图13的(d)示出了根据一个示例的当正向LFNST的输出数据的数量的最大值被限制为16时,对未应用4×4LFNST的其余块执行清零。

[0246] (ii)-(B)当应用8×8LFNST时,未应用8×8LFNST的区域可以被清零。图14是例示根据示例的应用8×8LFNST的块中的清零的图。

[0247] 如图14所示,关于应用8×8LFNST的块,即,对于图12的(d)和(e)中的所有块,LFNST未被应用到的整个区域可以用零填充。

[0248] (iii)由于以上(ii)中提出的清零,当应用LFNST时,用零填充的区域可以是不同的。因此,根据(ii)中提出的清零,与图12的LFNST的情况相比可以在更宽的区域上检查是否存在非零数据。

[0249] 例如,当应用(ii)-(B)时,除了图12的(d)和(e)中用零填充的区域之外,在图14中附加地用0填充的区域中检查是否存在非零数据之后,可以仅当不存在非零数据时发信号通知LFNST索引。

[0250] 当然,即使应用(ii)中提出的清零,也可以与现有LFNST索引信令以相同的方式检查是否存在非零数据。也就是说,在检查在图12中用零填充的块中是否存在非零数据之后,可以应用LFNST索引信令。在此情况下,编码设备仅执行清零并且解码设备不假设该清零,也就是说,仅检查非零数据是否仅存在于图12中显式标记为0的区域中,可以执行LFNST索引解析。

[0251] 可以推导出应用LFNST的简化方法((i)、(ii)-(A)、(ii)-(B)、(iii))的组合的各种实施方式。当然,上述简化方法的组合不限于以下实施方式,并且可以将任何组合应用于LFNST。

[0252]

[0253] -将正向LFNST的输出数据的数量限制为最大值16→(i)

[0254] -当应用4×4LFNST时,未应用4×4LFNST的所有区域被清零→(ii)-(A)

[0255] -当应用8×8LFNST时,未应用8×8LFNST的所有区域被清零→(ii)-(B)

[0256] -在检查非零数据是否也存在于填充有零值的现有区域以及由于附加的清零而填充有零的区域中((ii)-(A)、(ii)-(B))之后,仅在不存在非零数据时发信号通知LFNST索引※(iii)。

[0257] 在实施方式的情况下,当应用LFNST时,可以存在非零输出数据的区域限于左上4×4区域的内部。更具体地,在图13的(a)和图14的(a)中,扫描顺序中的第八个位置是其中可以存在非零数据的最后位置。在图13的(b)和(d)以及图14的(b)中,扫描顺序中的第十六个位置(即,左上4×4块的右下位置)是其中可以存在非零数据的最后位置。

[0258] 因此,在应用LFNST之后,在检查非零数据是否存在于残差编码过程不允许的位置(在超出最后位置的位置处)之后,可以确定是否发信号通知LFNST索引。

[0259] 在(ii)中提出的清零方法的情况下,由于当应用了一次变换和LFNST两者时最终生成的数据的数量,所以执行整个变换过程所需的计算量可以降低。也就是说,当LFNST被应用时,由于清零被应用于正向一次变换输出数据存在于未应用LFNST的区域中,因此不需要为在执行正向一次变换期间成为清零的区域生成数据。因此,可以降低生成对应的数据所需的计算量。在(ii)中提出的清零方法的附加效果总结如下。

[0260] 首先,如上所述,减少执行整个变换过程所需的计算量。

[0261] 特别地,当应用(ii)-(B)时,最坏情况的计算量被减少,使得变换过程可以被变轻。换句话说,一般来说,需要大量的计算来执行大尺寸的一次变换。通过应用(ii)-(B),作为执行正向LFNST的结果而推导出的数据的数量可以减小到16或更小。另外,随着整个块(TU或CU)的大小增加,减少变换操作的量的效果进一步增加。

[0262] 第二,可以减少整个变换过程所需的计算量,由此降低执行变换所需的功耗。

[0263] 第三,减小了变换过程中涉及的延迟。

[0264] 诸如LFNST之类的二次变换向现有的一次变换添加了计算量,因而增加了执行变换时涉及的总体延迟时间。特别地,在帧内预测的情况下,由于在预测过程中使用相邻块的重构数据,所以在编码期间,由于二次变换导致的延迟的增加导致直到重构的延迟的增加。这可以导致帧内预测编码的总体延迟的增加。

[0265] 然而,如果应用(ii)中提出的清零,则当应用LFNST时可以极大地减少执行一次变换的延迟时间,保持或减少整个变换的延迟时间,使得可以更简单地实现编码设备。

[0266] 此外,在传统的帧内预测中,编码目标块被视为一个编码单元,并且在不分割的情况下执行编码。然而,帧内子分区(ISP)编码意味着通过在水平方向或垂直方向上划分当前要编码的块来执行帧内预测编码。在这种情况下,可以通过以划分块为单位执行编码/解码来生成重构块,并且重构块可以用作下一个划分块的参考块。根据实施方式,在ISP编码中,可以将一个编码块划分为两个或四个子块并进行编码,并且在ISP中,在一个子块中,参考位于相邻的左侧或相邻的上侧的子块的重构像素值执行帧内预测。在下文中,“编码”可以用作包括由编码设备执行的编码和由解码设备执行的解码两者的概念。

[0267] ISP是将预测为亮度帧内的块根据块的大小在垂直方向或水平方向上划分为两个或四个子分区。例如,可以应用ISP的最小块大小为4×8或8×4。当块大小大于4×8或8×4时,将块划分为4个子分区。

[0268] 当应用ISP时,子块根据划分类型从左到右或从上到下顺序地进行编码(例如,水平地或垂直地),并在经由针对一个子块的逆变换和帧内预测执行重构处理之后,可以执行下一个子块的编码。对于最左或最上的子块,参考已经编码的编码块的重构像素,如在传统的帧内预测方法中一样。此外,当后续内部子块的每一侧不与前一子块相邻时,为了推导与对应侧相邻的参考像素,参考已编码的相邻编码块的重构像素,如在传统的帧内预测方法中一样。

[0269] 在ISP编码模式中,可以用相同的帧内预测模式对所有子块进行编码,并且可以发信号通知指示是否使用ISP编码的标志以及指示执行划分的方向(水平或垂直)的标志。此时,子块的数量可以依据块形状调整为2或4,并且当一个子块的大小(宽度×高度)小于16时,针对对应的子块可以不允许划分,或者可以设置限制ISP编码本身的应用。

[0270] 此外,在ISP预测模式的情况下,将一个编码单元划分为两个或四个分区块(即,子块)并进行预测,并且将相同的帧内预测模式应用于划分的两个或四个分区块。

[0271] 如上所述,在划分方向上,水平方向(当分别具有M和N的水平长度和垂直长度的M×N编码单元在水平方向上进行划分时,如果M×N编码单元被划分为两个,则M×N编码单元被划分为M×(N/2)块,并且如果M×N编码单元被划分为四个块,则M×N编码单元被划分为M×(N/4)块)和垂直方向(当M×N编码单元在垂直方向上进行划分时,如果M×N编码单元被划分为两个,则M×N编码单元被划分为(M/2)×N块,并且如果M×N编码单元被划分为四个,

则M×N编码单元被划分为(M/4)×N块)二者都是可能的。当在水平方向上划分M×N编码单元时,按照上下顺序对分区块进行编码,并且当在垂直方向上划分M×N编码单元时,按照左右顺序对分区块进行编码。在水平(垂直)方向划分的情况下,可以参考上(左)分区块的重构像素值来预测当前编码的分区块。

[0272] 可以对通过ISP预测方法以分区块为单位生成的残差信号应用变换。基于DST-7/DCT-8组合以及现有的DCT-2的多变换选择(MTS)技术可以应用于基于正向的一次变换(核心变换),并且正向低频不可分离的变换(LFNST)可以应用于根据一次变换生成的变换系数以生成最终修改的变换系数。

[0273] 也就是说,LFNST可以应用于通过应用ISP预测模式划分成的分区块,并且相同的帧内预测模式应用于划分的分区块,如上所述。因此,当选择基于帧内预测模式推导的LFNST集时,所推导的LFNST集可以应用于所有的分区块。也就是说,因为相同的帧内预测模式应用于所有分区块,所以相同的LFNST集可以应用于所有分区块。

[0274] 此外,根据示例,LFNST可以仅应用于具有4或更大的水平长度和垂直长度二者的变换块。因此,当根据ISP预测方法的划分的分区块的水平长度或垂直长度小于4时,不应用LFNST并且不发信号通知LFNST索引。此外,当对每个分区块应用LFNST时,可以将对应的分区块视为一个变换块。当然,当不应用ISP预测方法时,可以将LFNST应用于编码块。

[0275] LFNST到每个分区块的应用将详细描述如下。

[0276] 根据示例,在将正向LFNST应用于各个分区块之后,在按变换系数扫描顺序的左上4×4区域中仅留下最多16个(8个或16个)系数,然后可以应用清零,其中其余的位置和区域全部被填充为0。

[0277] 另选地,根据示例,当分区块的一侧的长度为4时,LFNST仅应用于左上4×4区域,并且当分区块的所有侧的长度(即,宽度和高度)为8或更大时,LFNST可以应用于左上8×8区域内的除了右下4×4区域之外的其余48个系数。

[0278] 另选地,根据示例,为了将最坏情况的计算复杂度调整为每个样本8次乘法,当每个分区块是4×4或8×8时,在应用正向LFNST之后可以仅输出8个变换系数。也就是说,当分区块为4×4时,可以应用8×16矩阵作为变换矩阵,当分区块为8×8时,可以应用8×48矩阵作为变换矩阵。

[0279] 此外,在当前的VVC标准中,以编码单元为单位执行LFNST索引信令。因此,在ISP预测模式中并且当对所有分区块应用LFNST时,可以将相同的LFNST索引值应用于对应的分区块。也就是说,当在编码单元级别发送一次LFNST索引值时,对应的LFNST索引可以应用于编码单元中的所有分区块。如上所述,LFNST索引值可以具有0、1和2的值,其中0表示不应用LFNST的情况,而1和2表示在应用LFNST时存在于一个LFNST集中的两个变换矩阵。

[0280] 如上所述,LFNST集是由帧内预测模式确定的,并且在ISP预测模式的情况下,因为编码单元中的所有分区块都是在相同的帧内预测模式下预测的,所以分区块可以参考相同的LFNST集。

[0281] 作为另一示例,LFNST索引信令仍然以编码单元为单位来执行,但是在ISP预测模式的情况下,不确定是否对所有分区块均一地应用LFNST,并且对于每个分区块,可以通过单独的条件来确定是否应用在编码单元级别发信号通知的LFNST索引值以及是否应用LFNST。这里,可以通过比特流以针对每个分区块的标志的形式发信号通知单独的条件,并

且当标志值为1时,应用在编码单元级别发信号通知的LFNST索引值,并且当标志值为0时,可以不应用LFNST。

[0282] 在下文中,将描述在将LFNST应用于ISP模式时保持最坏情况的计算复杂度的方法。

[0283] 在ISP模式的情况下,当应用LFNST时,为了将每个样本(或每个系数、每个位置)的乘法次数保持在某个值或更少,可能会限制LFNST的应用。根据分区块的大小,通过如下应用LFNST,每个样本(或每个系数,每个位置)的乘法次数可以保持为8或更少。

[0284] 1.当分区块的水平长度和垂直长度二者都为4或更大时,可以应用与当前VVC标准中针对LFNST的最坏情况的计算复杂度控制方法相同的方法。

[0285] 也就是说,当分区块是4×4块时,可以在正向方向上应用通过从16×16矩阵中采样上8行获得的8×16矩阵而不是16×16矩阵,并且可以在逆方向上应用通过从16×16矩阵中采样左8列获得的16×8矩阵。此外,当分区块是8×8块时,在正向方向上,代替16×48矩阵,应用通过从16×48矩阵中采样上8行而获得的8×48矩阵,并且在逆方向上,代替48×16矩阵,可以应用通过从48×16矩阵中采样左8列获得的48×8矩阵。

[0286] 在4×N或N×4(N>4)块的情况下,当执行正向变换时,在将16×16矩阵仅应用于左上4×4块之后生成的16个系数可以设置在左上4×4区域中,并且其它区域可以填充有值0。另外,在执行逆变换时,将位于左上4×4块中的16个系数按扫描顺序设置以形成输入向量,然后可以通过乘以16×16矩阵生成16个输出数据。生成的输出数据可以设置在左上4×4区域中,并且除了左上4×4区域之外的其余区域可以填充有值0。

[0287] 在8×N或N×8(N>8)块的情况下,当执行正向变换时,在将16×48矩阵应用于仅左上8×8块内的ROI区域(除了来自左上8×8块中的右下4×4块之外的其余区域)后的生成16个系数可以设置在左上4×4区域中,并且所有其它区域可以填充有值0。此外,在执行逆变换时,位于左上4×4区域中的16个系数按扫描顺序设置以形成输入向量,然后可以通过乘以48×16矩阵生成48个输出数据。生成的输出数据可以填充在ROI区域中,并且所有其它区域可以填充有值0。

[0288] 作为另一示例,为了将每个样本(或每个系数,每个位置)的乘法次数保持在某个值或更少,基于ISP编码单元大小而不是ISP分区块的大小的每个样本(或每个系数,每个位置)的乘法次数可以保持为8或更少。当ISP分区块当中只有一个块满足应用LFNST的条件时,可以基于对应的编码单元大小而不是分区块的大小来应用LFNST的最坏情况的复杂度计算。例如,如果特定编码单元的亮度编码块通过被分割成具有4×4大小的四个分区块而用ISP进行编码,并且对于它们当中的两个分区块不存在非零变换系数,则它可以被配置为使得针对其它两个分区块中的每个分区块不是生成8个,而是生成16个变换系数。

[0289] 在下文中,将描述在ISP模式的情况下发信号通知LFNST索引的方法。

[0290] 如上所述,LFNST索引可以具有值0、1和2,其中0指示未应用LFNST,并且1和2指示包括在所选择的LFNST集中的两个LFNST核矩阵中的一个。在当前的VVC标准中发送LFNST索引的方法将描述如下。

[0291] 1.可以针对每个编码单元(CU)发送一次LFNST索引。在双树情况下,可针对亮度块和色度块中的每一者发信号通知单独的LFNST索引。

[0292] 2.当未发信号通知LFNST索引时,将LFNST索引推断为作为默认值的0。在下面的情

况下LFNST索引值被推断为0。

[0293] A.当在不应用变换的模式(例如,变换跳过、BDPCM、无损编码等)中时

[0294] B.当一次变换不是DCT-2时(当它是DST7或DCT8时),即,当水平变换或垂直变换不是DCT-2时

[0295] C.当编码单元的亮度块的水平长度或垂直长度超过可变换最大亮度变换的大小时,例如,当在可变换最大亮度变换的大小为64的情况下编码块的亮度块的大小为128×16时,LFNST不适用。

[0296] 在双树情况下,确定亮度分量的编码单元和色度分量的编码单元中的每一者是否超过最大亮度变换的大小。也就是说,检查亮度块是否超过可变换最大亮度变换的大小,并且检查色度块是否超过用于颜色格式的对应亮度块的水平/垂直长度和最大可变换亮度变换的大小。例如,当颜色格式为4:2:0时,对应亮度块的水平/垂直长度中的每一者是色度块的水平/垂直长度的两倍,并且对应亮度块的变换大小是色度块的变换大小的两倍。作为另一示例,当颜色格式为4:4:4时,对应亮度块的水平/垂直长度与色度块的水平/垂直长度相同。

[0297] 64-长度变换或32-长度变换可表示分别以64或32的长度水平或垂直应用的变换,并且“变换大小”可以表示64或32的对应长度。

[0298] 在单树情况下,检查亮度块的水平或垂直长度是否超过可变换最大亮度变换块的大小,并且如果其超过该大小,则可以跳过LFNST索引信令。

[0299] D.可仅在编码单元的水平长度和垂直长度两者大于或等于4时发送LFNST索引。

[0300] 在双树情况下,可仅针对对应分量(即,亮度或色度分量)的水平长度和垂直长度两者大于或等于4的情况发信号通知LFNST索引。

[0301] 在单树情况下,可仅针对亮度分量的水平长度和垂直长度两者大于或等于4的状况发信号通知LFNST索引。

[0302] E.在最后非零系数位置不是DC位置(块的左上位置)的情况下,当它是双树类型亮度块时,如果最后非零系数位置不是DC位置,则发送LFNST索引。当它是双树类型色度块时,如果即使Cb的最后非零系数位置和Cr的最后非零系数位置中的一者不是DC位置,则发送LNFST索引。

[0303] 在单树类型的情况下,如果即使在亮度分量、Cb分量和Cr分量中的一者中最后非零系数位置不是DC位置,则发送LFNST索引。

[0304] 这里,如果指示针对一个变换块是否存在变换系数的编码块标志(CBF)值为0,则不检查变换块的最后非零系数位置来确定是否发信号通知LFNST索引。也就是说,如果CBF值是0,则由于变换不应用于块,所以当检查用于LFNST索引信令的条件时,可以不考虑最后非零系数位置。

[0305] 例如,1)在双树类型和亮度分量的情况下,如果对应CBF值是0,则不发信号通知LFNST索引,2)在双树类型和色度分量的情况下,如果Cb的CBF值是0且Cr的CBF值是1,则通过仅检查Cr的最后非零系数位置来发送LFNST索引,并且3)在单树类型的情况下,对于亮度、Cb和Cr中的全部,仅针对相应CBF值是1的分量检查最后非零系数位置。

[0306] F.当确认在除了可以存在LFNSF变换系数的位置之外的位置处存在变换系数时,可以跳过LFNST索引信令。在4×4变换块和8×8变换块的情况下,LFNST变换系数可以根据

VVC标准中的变换系数扫描顺序从DC位置开始存在于8个位置,并且所有其余位置用0填充。另外,当变换块不是4×4变换块和8×8变换块时,LFNST变换系数可以根据VVC标准中的变换系数扫描顺序从DC位置开始存在于16个位置处,并且所有其余位置用0填充。

[0307] 因此,如果在执行残差编码之后非零变换系数存在于要用0填充的区域中,则可以省略LFNST索引信令。

[0308] 此外,ISP模式可仅应用于亮度块或可应用于亮度块和色度块两者。如上文所描述,当应用ISP预测时,通过将对应编码单元分割成2个或4个分区块来进行预测,并且可以将变换应用于每个分区块。因此,当在编码单元的基础上确定其中发信号通知LFNST索引的条件时,也应考虑LFNST适用于每个分区块的事实。另外,当将ISP预测模式仅应用于特定分量(例如,亮度块)时,应通过考虑仅针对分量实现分割成分区块的事实来发信号通知LFNST索引。当处于ISP模式时可能的LFNST索引信令方案总结如下。

[0309] 1.可以针对每个编码单元(CU)发送一次LFNST索引。在双树情况下,可针对亮度块和色度块中的每一者发信号通知单独的LFNST索引。

[0310] 2.当未发信号通知LFNST索引时,将LFNST索引推断为作为默认值的0。在下面的情况下LFNST索引值被推断为0。

[0311] A.当在不应用变换的模式(例如,变换跳过、BDPCM、无损编码等)中时

[0312] B.当编码单元的亮度块的水平长度或垂直长度超过可变换最大亮度变换的大小时,例如,当在可变换最大亮度变换的大小为64的情况下编码块的亮度块的大小为128×16时,LFNST不适用。

[0313] 还可以基于分区块而非编码单元的大小来确定是否发信号通知LFNST索引。也就是说,当对应亮度块的分区块的水平长度或垂直长度超过可变换最大亮度变换的大小时,可以跳过LFNST索引信令,并且可以推断LFNST索引为0。

[0314] 在双树情况下,确定亮度分量的编码单元或分区块和色度分量的编码单元或分区块中的每一者是否超过最大变换块大小。也就是说,将亮度的编码单元或分区块的水平和垂直长度中的每一个与最大亮度变换尺寸进行比较,并且如果它们中的任何一个大于最大亮度变换尺寸,则不应用LFNST,并且在色度的编码单元或分区块的情况下,用于颜色格式的对应亮度块的水平/垂直长度与可变换最大亮度变换尺寸进行比较。例如,当颜色格式为4:2:0时,对应亮度块的水平/垂直长度中的每一者是色度块的水平/垂直长度的两倍,并且对应亮度块的变换大小是色度块的变换大小的两倍。作为另一示例,当颜色格式为4:4:4时,对应亮度块的水平/垂直长度与色度块的水平/垂直长度相同。

[0315] 在单树情况下,检查亮度块(编码单元或分区块)是否超过可变换最大亮度变换块的大小,并且如果其超过该大小,则可跳过LFNST索引信令。

[0316] C.如果应用了当前VVC标准中包括的LFNST,则可以仅当分区块的水平长度和垂直长度二者都大于或等于4时才发送LFNST索引。

[0317] 如果除了当前VVC标准中包括的LFNST之外还应用了针对2×M(1×M)或M×2(M×1)块的多达LFNST,则仅当分区块具有大于或等于2×M(1×M)或M×2(M×1)块的大小时才发送LFNST索引。这里,P×Q块大于或等于R×S块表示P≥R并且Q≥S。

[0318] 总之,可以仅当分区块具有大于或等于适用于LFNST的最小大小的大小时才发送LFNST索引。在双树情况下,可仅在亮度或色度分量的分区块具有大于或等于适用于LFNST

的最小大小的大小时发信号通知LFNST索引。在单树情况下,可仅在亮度分量的分区块具有大于或等于适用于LFNST的最小大小的大小时发信号通知LFNST索引。

[0319] 在本文档中,M×N块等于或大于K×L块表示M等于或大于K并且N等于或大于L。M×N块大于K×L块表示M等于或大于K并且N等于或大于L,其中M大于K或者N大于L。M×N块小于或等于K×L块表示M小于或等于K并且N小于或等于L。M×N块小于K×L块表示M小于或等于K并且N小于或等于L,其中M小于K或者N小于L。

[0320] D.在最后非零系数位置不是DC位置(块的左上位置)的情况下,当它是双树类型亮度块时,如果对应的最后非零系数位置不是所有分区块中的甚至任何一个中的DC位置,则可以执行LFNST发送。当它是双树类型和色度块时,如果Cb的所有分区块的非零系数位置(这里,当ISP模式没有应用于色度分量时,认为分区块的数量是1)和Cr的所有分区块的最后非零系数位置(这里,当ISP模式没有应用于色度分量时,认为分区块的数量是1)中的甚至任何一者不是DC位置,则可以发送对应的LNFST索引。

[0321] 在单树类型的情况下,如果即使在亮度分量、Cb分量和Cr分量的所有分区块中的任何一个中,最后非零位置不是DC位置,则可发送对应的LNFST索引。

[0322] 这里,如果指示针对每个分区块是否存在变换系数的编码块标志(CBF)值为0,则不检查对应分区块的最后非零系数位置来确定是否发信号通知LFNST索引。也就是说,如果CBF值是0,则由于不将变换应用于块,所以当检查用于LFNST索引信令的条件时不考虑最后非零系数位置。

[0323] 例如,1)在双树类型和亮度分量的情况下,当确定是否针对每个分区块发信号通知LFNST索引时,如果对应CBF值是0,2)在双树类型和色度分量的情况下,如果Cb的CBF值为0且Cr的CBF值为1,则可以通过仅检查每个分区块的Cr的非零系数位置来确定是否发信号通知LFNST索引,以及3)在单树类型的情况下,可以通过仅针对亮度分量、Cb分量和Cr分量的所有分区块中CBF值为1的块检查最后非零系数位置来确定是否发信号通知LFNST索引。

[0324] 在ISP模式的情况下,图像信息可以被配置为不检查最后的非零系数位置,并且其实施方式可以如下。

[0325] i.在ISP模式的情况下,可以通过跳过对亮度块和色度块两者的非零系数位置的检查来允许LFNST索引信令。也就是说,即使最后非零系数位置对于所有分区块都是DC位置或对应CBF值是0,也可允许LFNST索引信令。

[0326] ii.在ISP模式的情况下,基于前述方案,可仅针对亮度块跳过对非零系数位置的检查,并且可以针对色度块检查非零系数位置。例如,在双树类型和亮度块的情况下,允许LFNST索引信令而不检查非零系数位置,并且在双树类型和色度块的情况下,可以通过检查通过前述方案对于最后非零系数位置是否存在DC位置来确定是否发信号通知LFNST索引。

[0327] iii.在ISP模式和单树类型的情况下,可以应用方案i或ii。也就是说,在ISP模式和单树类型的情况下,当应用方案i时,可以通过跳过对亮度块和色度块两者的最后非零系数位置的检查来允许LFNST索引信令。另选地,可应用方案ii以通过上述方案按照针对亮度分量的分区块跳过针对非零系数位置的检查且针对色度分量的分区块执行针对非零系数位置的检查的方式来确定发信号通知LFNST索引(当ISP未应用于色度分量时,可以将分区块的数量视为1)。

[0328] E.当确认即使针对所有分区块当中的一个分区块,变换系数存在于除了可以存在

LFNST变换系数的位置之外的位置时,可以省略LFNST索引信令。

[0329] 例如,在4×4分区块和8×8分区块的情况下,LFNST变换系数可以根据VVC标准中的变换系数扫描顺序存在于从DC位置开始的8个位置处,并且所有其余位置用0填充。另外,当变换块大于或等于4×4大小并且不是4×4分区块和8×8分区块时,LFNST变换系数可以根据VVC标准中的变换系数扫描顺序存在于从DC位置开始的16个位置处,并且所有其余位置用0填充。

[0330] 因此,如果在执行残差编码之后非零变换系数存在于要用0填充的区域中,则可以省略LFNST索引信令。

[0331] 此外,在ISP模式的情况下,代替DCT-2,通过针对水平方向和垂直方向中的每一个独立地考虑长度条件,在没有针对当前VVC标准中的MTS索引的信令的情况下应用DST-7。确定水平或垂直长度是否大于或等于4或大于或等于16,并且根据确定的结果确定一次变换核。因此,当处于ISP模式时并且当LFNST适用时,可以如下配置变换组合。

[0332] 1.在LFNST索引为0的情况下(包括LFNST索引被推断为0的情况),当ISP被包括在当前VVC标准中时,其可符合确定一次变换的条件。也就是说,检查针对水平方向和垂直方向中的每一者是否独立地满足长度条件(长度大于或等于4且小于或等于16的条件),并且如果满足条件,则针对一次变换应用DST-7而非DCT-2,并且如果不满足条件,则可应用DCT-2。

[0333] 2.在LFNST索引大于0的情况下,对于一次变换,以下两种配置是可能的。

[0334] A.DCT-2适用于水平方向和垂直方向两者。

[0335] B.当ISP被包括在当前VVC标准中时,其可符合确定一次变换的条件。也就是说,检查针对水平方向和垂直方向中的每一者是否独立地满足长度条件(长度大于或等于4且小于或等于16的条件),并且如果满足条件,则应用DST-7而非DCT-2,并且如果不满足条件,则可应用DCT-2。

[0336] 在ISP模式中,图像信息可以被配置为使得LFNST索引不是每个编码单元而是每个分区块地发送。在这种情况下,可以通过认为在前述LFNST索引信令方案中发送LFNST索引的单元中仅存在一个分区块来确定是否发信号通知LFNST索引。

[0337] 在下文中,描述了LFNST索引和MTS索引的信令。

[0338] 根据示例的与LFNST索引和MTS索引的信令相关的编码单元语法表、变换单元语法表和残差编码语法表如下所示。根据表4,MTS索引从变换单元级别语法移动到编码单元级别语法,并在发信号通知LFNST索引之后被发信号通知。此外,去除了当将ISP应用于编码单元时不允许LFNST的约束。由于去除了当将ISP应用于编码单元时不允许LFNST的约束,LFNST可以应用于所有帧内预测块。此外,在编码单元级别的结尾处有条件地发信号通知MTS索引和LFNST索引二者。

[0339] [表4]

[0340]

```
coding_unit( x0, y0, cbWidth, cbHeight, cqtDepth, treeType, modeType ) {
…
    LfnstDcOnly = 1
    LfnstZeroOutSigCoeffFlag = 1
    MtsZeroOutSigCoeffFlag = 1
    transform_tree( x0, y0, cbWidth, cbHeight, treeType )
    lfnstWidth = ( treeType == DUAL_TREE_CHROMA ) ? cbWidth / SubWidthC
                 : ( IntraSubPartitionsSplitType ==
ISP_VER_SPLIT) ? cbWidth / NumIntraSubPartitions : cbWidth
    lfnstHeight = ( treeType == DUAL_TREE_CHROMA ) ? cbHeight / SubHeightC
                  : ( IntraSubPartitionsSplitType ==
ISP_HOR_SPLIT) ? cbHeight / NumIntraSubPartitions : cbHeight
    if( Min( lfnstWidth, lfnstHeight ) >= 4 && sps_lfnst_enabled_flag == 1 &&
      CuPredMode[ chType ][ x0 ][ y0 ] == MODE_INTRA &&
      ( !intra_mip_flag[ x0 ][ y0 ] || Min( lfnstWidth, lfnstHeight ) >= 16 ) &&
    Max( cbWidth, cbHeight ) <= MaxTbSizeY) {
      if( ( IntraSubPartitionsSplitType != ISP_NO_SPLIT || LfnstDcOnly == 0 ) &&
LfnstZeroOutSigCoeffFlag == 1 )
        lfnst_idx[ x0 ][ y0 ]
    }
    if( treeType != DUAL_TREE_CHROMA && lfnst_idx[ x0 ][ y0 ] == 0 &&
      transform_skip_flag[ x0 ][ y0 ] == 0 && Max( cbWidth, cbHeight ) <= 32 &&
      IntraSubPartitionsSplit[ x0 ][ y0 ] == ISP_NO_SPLIT && ( !cu_sbt_flag ) &&
      MtsZeroOutSigCoeffFlag == 1 && tu_cbf_luma[ x0 ][ y0 ] ) {
      if( ( ( CuPredMode[ chType ][ x0 ][ y0 ] == MODE_INTER &&
        sps_explicit_mts_inter_enabled_flag )
        || ( CuPredMode[ chType ][ x0 ][ y0 ] == MODE_INTRA &&
        sps_explicit_mts_intra_enabled_flag ) ) )
        mts_idx[ x0 ][ y0 ]
    }
…
```

[0341] [表5]

[0342]

```
transform_unit( x0, y0, tbWidth, tbHeight, treeType, subTuIndex, chType ) {
…
  if( tu_cbf_luma[ x0 ][ y0 ] && treeType != DUAL_TREE_CHROMA
    && ( tbWidth <= 32 ) && ( tbHeight <= 32 )
    && ( IntraSubPartitionsSplit[ x0 ][ y0 ] == ISP_NO_SPLIT ) && ( !cu_sbt_flag ) ) {
    if( sps_transform_skip_enabled_flag && !BdpcmFlag[ x0 ][ y0 ] &&
      tbWidth <= MaxTsSize && tbHeight <= MaxTsSize )
      transform_skip_flag[ x0 ][ y0 ]
  }
…
```

[0343] [表6]

[0344]

```
residual_coding( x0, y0, log2TbWidth, log2TbHeight, cIdx ) {
…
   if( ( sps_mts_enabled_flag   &&
cu_sbt_flag && log2TbWidth < 6 && log2TbHeight < 6 ) )
         &&   cIdx = = 0   &&   log2TbWidth > 4 )
      log2ZoTbWidth = 4
   else
      log2ZoTbWidth = Min( log2TbWidth, 5 )
   MaxCcbs = 2 * ( 1 << log2TbWidth ) * ( 1<< log2TbHeight )
   if( ( sps_mts_enabled_flag   &&
cu_sbt_flag && log2TbWidth < 6 && log2TbHeight < 6 ) )
         &&   cIdx = = 0   &&   log2TbHeight > 4 )
      log2ZoTbHeight = 4
   else
      log2ZoTbHeight = Min( log2TbHeight, 5 )
…
   if( ( lastSubBlock > 0   &&   log2TbWidth >= 2   &&   log2TbHeight >= 2 )   ||
      ( lastScanPos > 7   &&   ( log2TbWidth = = 2   ||   log2TbWidth = = 3 )   &&
      log2TbWidth = = log2TbHeight ) )
      LfnstZeroOutSigCoeffFlag = 0
   if( ( LastSignificantCoeffX > 15   ||   LastSignificantCoeffY > 15 ) && cIdx = = 0 )
      MtsZeroOutSigCoeffFlag = 0
…
```

[0345] 表中所示的主要变量的含义如下。

[0346] 1.cbWidth、cbHeight:当前编码块的宽度和高度

[0347] 2.log2TbWidth、log2TbHeight:当前变换块的宽度和高度的以2为底的对数值,其可以通过反映清零而减小到可以存在非零系数的左上区域。

[0348] 3.sps_lfnst_enabled_flag:指示是否启用LFNST的标志,如果标志值为0,则它指示不启用LFNST,并且如果标志值为1,则它指示启用LFNST。它被定义在序列参数集(SPS)中。

[0349] 4.CuPredMode[chType][x0][y0]:与变量chType和(x0,y0)位置相对应的预测模式,chType可以具有0和1的值,其中0指示亮度分量,并且1指示色度分量。(x0,y0)位置指示图片上的位置,并且MODE_INTRA(帧内预测)和MODE_INTER(帧间预测)可以作为CuPredMode[chType][x0][y0]的值。

[0350] 5.IntraSubPartitionsSplit[x0][y0]:(x0,y0)位置的内容与第4条中相同。它指示在(x0,y0)位置应用了哪个ISP分区,ISP_NO_SPLIT指示对应于(x0,y0)位置的编码单元没有被划分为分区块。

[0351] 6.intra_mip_flag[x0][y0]:(x0,y0)位置的内容与以上第4条中相同。intra_mip_flag是指示是否应用基于矩阵的帧内预测(MIP)预测模式的标志。如果标志值为0,则它指示未启用MIP,如果标志值为1,则它指示启用了MIP。

[0352] 7.cIdx:值0指示亮度,并且值1和2指示分别是色度分量的Cb和Cr。

[0353] 8.treeType:指示单树和双树等(SINGLE_TREE:单树,DUAL_TREE_LUMA:用于亮度

分量的双树,DUAL_TREE_CHROMA:用于色度分量的双树)

[0354] 9.tu_cbf_cb[x0][y0]:(x0,y0)位置的内容与第4条中相同。它指示Cb分量的编码块标志(CBF)。如果其值为0,则它意味着在Cb分量的对应变换单元中不存在非零系数,并且如果其值为1,则它指示Cb分量的对应变换单元中存在非零系数。

[0355] 10.lastSubBlock:它指示最后非零系数所位于的子块(系数组(CG))按照扫描顺序的位置。0指示包含DC分量的子块,并且在大于0的情况下,它不是包含DC分量的子块。

[0356] 11.lastScanPos:它指示在一个子块内最后有效系数按照扫描顺序所在的位置。如果一个子块包括16个位置,则可以有从0至15的值。

[0357] 12.lfnst_idx[x0][y0]:要解析的LFNST索引语法元素。如果不解析,则它被推断为值0。也就是说,默认值被设置为0,指示不应用LFNST。

[0358] 13.cu_sbt_flag:指示当前VVC标准中包括的子块变换(SBT)是否适用的标志。标志值等于0指示SBT不适用,并且标志值等于1指示应用SBT。

[0359] 14.sps_explicit_mts_inter_enabled_flag、sps_explicit_mts_intra_enabled_flag:分别指示显式MTS是否应用于帧间CU和帧内CU的标志。标志值等于0指示MTS不适用于帧间CU或帧内CU,并且标志值等于1指示MTS适用。

[0360] 15.tu_mts_idx[x0][y0]:要解析的MTS索引语法元素。当不解析时,该元素被推断为值0。也就是说,该元素被设置为默认值0,该默认值0指示水平地和垂直地都应用DCT-2。

[0361] 如表4所示,当对mts_idx[x0][y0]进行编码时检查多个条件,并且仅在lfnst_idx[x0][y0]等于0时才发信号通知tu_mts_idx[x0][y0]。

[0362] tu_cbf_luma[x0][y0]是指示对于亮度分量是否存在有效系数的标志。

[0363] 根据表4,当亮度分量的编码单元的宽度和高度二者都为32或更小时,发信号通知mts_idx[x0][y0](Max(cbWidth,cbHeight)<=32),也就是说,根据亮度分量的编码单元的宽度和高度来确定是否应用MTS。

[0364] 此外,根据表4,可以配置为即使在ISP模式(IntraSubPartitionsSplitType!=ISP_NO_SPLIT)下也发信号通知lfnst_idx[x0][y0],并且相同的LFNST索引值可以应用于所有ISP分区块。

[0365] 然而,仅在除了ISP模式之外的情况(IntraSubPartitionsSplit[x0][y0]==ISP_NO_SPLIT)下发信号通知mts_idx[x0][y0]。

[0366] 如表6所示,在确定log2ZoTbWidth和log2ZoTbHeight的过程中(其中,log2ZoTbWidth和log2ZoTbHeight分别表示在执行清零之后剩余的左上区域的宽度和高度的以2为底的对数值),可以省略检查mts_idx[x0][y0]的值。

[0367] 根据示例,当在残差编码中确定log2ZoTbWidth和log2ZoTbHeight时,可以添加检查sps_mts_enable_flag的条件。

[0368] 如果在应用LFNST时在清零位置处存在有效系数,则表4中的变量LfnstZeroOutSigCoeffFlag为0,否则为1。可以根据表6所示的多个条件来设置变量LfnstZeroOutSigCoeffFlag。

[0369] 根据示例,表4中的变量LfnstDcOnly在对应的编码块标志(CBF,当在对应块中存在至少有一个有效系数时为1,否则为0)为1的变换块的所有最后有效系数在DC位置(左上位置)时为1,否则为0。具体而言,在双树亮度中针对一个亮度变换块检查最后有效系数的

位置,并且在双树色度中针对Cb的变换块和Cr的变换块二者检查最后有效系数的位置。在单树中,可以针对亮度、Cb和Cr的变换块检查最后有效系数的位置。

[0370] 在表4中,MtsZeroOutSigCoeffFlag初始被设置为1,并且在表6的残差编码中可以改变该值。当在通过清零要用0填充的区域(LastSignificantCoeffX>15||LastSignificantCoeffY>15)中存在有效系数时,变量MtsZeroOutSigCoeffFlag的值从1变为0,在这种情况下,不发信号通知MTS索引,如表4所示。

[0371] 如表4所示,当tu_cbf_luma[x0][y0]为0时,可以省略mts_idx[x0][y0]编码。也就是说,当亮度分量的CBF值为0时,没有应用变换,因此不需要发信号通知MTS索引。因此,可以省略MTS索引编码。

[0372] 根据示例,以上技术特征可以用另一种条件语法来实现。例如,在执行MTS之后,可以推导指示在当前块的除了DC区域之外的区域中是否存在有效系数的变量,并且当该变量指示在除了DC区域之外的区域中存在有效系数时,可以发信号通知MTS索引。也就是说,在当前块的除了DC区域之外的区域中存在有效系数指示tu_cbf_luma[x0][y0]的值为1,并且在这种情况下,可以发信号通知MTS索引。

[0373] 该变量可以表示为MtsDcOnly,并且在变量MtsDcOnly在编码单元级别被初始设置为1之后,在残差编码级别中当确定在当前块的除了DC区域之外的区域中存在有效系数时,该值被改变为0。当变量MtsDcOnly为0时,图像信息可以被配置使得发信号通知MTS索引。

[0374] 当tu_cbf_luma[x0][y0]为0时,由于在表5的变换单元级别没有调用残差编码语法,因此保持变量MtsDcOnly的初始值1。在这种情况下,由于变量MtsDcOnly没有被改变为0,所以图像信息可以被配置为不发信号通知MTS索引。也就是说,不解析和发信号通知MTS索引。

[0375] 此外,解码设备可以确定变换系数的颜色索引cIdx,以推导表6的变量MtsZeroOutSigCoeffFlag。颜色索引cIdx为0表示亮度分量。

[0376] 根据示例,由于MTS可以仅应用于当前块的亮度分量,因此解码设备在推导用于确定是否解析MTS索引的变量MtsZeroOutSigCoeffFlag时,可以确定颜色索引是否为亮度。

[0377] 变量MtsZeroOutSigCoeffFlag是指示当应用MTS时是否执行清零的变量。其指示在执行MTS之后由于清零可以存在最后有效系数的左上区域之外的区域(即,除了左上16×16区域之外的区域)中是否存在变换系数。变量MtsZeroOutSigCoeffFlag在编码单元级别初始被设置为1,如表4所示(MtsZeroOutSigCoeffFlag=1),并且当在除了16×16区域之外的区域中存在变换系数时,其值可以在残差编码级别从1改变为0,如表6所示(MtsZeroOutSigCoeffFlag=0)。当变量MtsZeroOutSigCoeffFlag的值为0时,不发信号通知MTS索引。

[0378] 如表6所示,在残差编码级别,可以依据是否执行伴随MTS的清零来设置其中可以存在非零变换系数的非清零区域,并且即使在这种情况下,颜色索引(cIdx)为0,非清零区域可以设置为当前块的左上16×16区域。

[0379] 如此,在推导确定是否解析MTS索引的变量时,确定颜色分量是亮度还是色度。然而,由于LFNST可以应用于当前块的亮度分量和色度分量二者,所以在推导用于确定是否解析LFNST索引的变量时不确定颜色分量。

[0380] 例如,表4示出了变量LfnstZeroOutSigCoeffFlag,其可以指示在应用LFNST时执

行清零。变量LfnstZeroOutSigCoeffFlag指示在当前块的除了左上的第一区域之外的第二区域中是否存在有效系数。该值初始被设置为1,并且当在第二区域中存在有效系数时,该值可以改变为0。仅在初始设置的变量LfnstZeroOutSigCoeffFlag的值保持为1时,可以解析LFNST索引。在确定并且推导变量LfnstZeroOutSigCoeffFlag的值是否为1时,由于LFNST可以应用于当前块的亮度分量和色度分量二者,因此不确定当前块的颜色索引。

[0381] 如表4所示,针对每个编码单元发信号通知MTS索引,并且仅在MtsZeroOutSigCoeffFlag等于1时才发信号通知mts_idx(MTS索引)。

[0382] 此外,如表6所示,可以在残差编码级别中根据特定条件(if((LastSignificantCoeffX>15||LastSignificantCoeffY>15)&&cIdx==0))来确定MtsZeroOutCoeffFlag的值。

[0383] 当最后非零系数的x坐标(LastSignificantCoeffX)大于15且最后非零系数的y坐标(LastSignificantCoeffY)大于15时,MtsZeroOutSigCoeffFlag的值设置为0。这里,x坐标在变换块中从左到右增加,并且y坐标在变换块中从上到下增加。变换块的左上位置是(0,0)。

[0384] 在表4中MtsZeroOutSigCoeffFlag的值被初始化为1,并且如果不满足表6中呈现的条件则保持为1,因此MtsZeroOutSigCoeffFlag的值不设置为0。

[0385] 当在水平方向应用的一次变换为A,并且在垂直方向应用的一次变换为B时,一次变换可以由(A,B)表示。在当前的VVC标准中,在(DST-7,DST-7)、(DST-7,DCT-8)、(DCT-8,DST-7)和(DCT-8,DCT-8)的情况下,变换系数被限制为仅存在于变换块的左上16×16区域中。因此,表6中(LastSignificantCoeffX>15||LastSignificantCoeffY>15)的条件指示最后非零系数位于左上16×16区域外部。在表6中,变量cIdx表示颜色分量,并且变量cIdx值为0表示亮度分量。

[0386] 在当前VVC标准中,在针对残差编码进行解析之后,解析MTS索引(mts_idx)。具体地,表4是作为编码单元的语法解析功能的编码单元,在变换树功能之后解析MTS索引,并且调用变换树功能中的解析残差编码的功能。因此,在解析残差之后解析MTS索引(表6所示的残差解析功能负责解析残差编码),并且由于在残差编码期间从解码的角度不知道MTS索引信息,所以可能不是无法知道应用了哪个一次变换。然而,当MTS适用并且应用SBT或ISP时,或者当应用隐式MTS时,隐式应用MTS,因此在解析残差编码之前知道一次变换也是可能的。

[0387] 表6所示的条件需要检查,因为当前VVC标准设置了当在32长度的DST-7或32长度的DCT-8中应用正向变换时仅输出16个变换系数的限制。例如,当MTS索引的值大于0并因此指示应用的一次变换是(DST-7,DST-7)、(DST-7,DCT-8)、(DCT-8,DST-7)和(DCT-8,DCT-8)之一时,非零变换系数可以仅存在于变换块的左上16×16区域中。

[0388] 因此,当最后非零变换系数的位置在对应于MTS索引值为0的情况的左上16×16区域外部时,可以省略MTS索引的信令,如表4所示。然而,即使最后非零变换系数的位置在左上16×16区域内,非零系数也可能存在于左上16×16区域外部。在这种情况下,即使MTS索引值为0,MTS索引也可以被发信号通知为0,这将在下面详细描述。

[0389] 当一次变换为(DCT-2,DCT-2)时,最后非零系数的位置存在于左上16×16区域,并且非零变换系数也可以存在于左上16×16区域之外,因为按照变换块内的扫描顺序,可以扫描左上16×16区域之外的区域(即,除了左上16×16区域之外的区域),然后可以扫描左

上16×16区域,或者可以扫描左上16×16区域然后可以扫描左上16×16区域之外的区域。

[0390] 图15例示了根据本公开的实施方式的32×32变换块的扫描,其具体示出了32×32变换块被划分为4×4系数组(CG)并被扫描。在本公开中,被划分以进行扫描的CG被称为“扫描子块”。

[0391] 针对每个CG所指示的数字表示正向扫描顺序,并且可以由解码设备从存在最后非零变换系数的位置开始按照逆扫描顺序来扫描CG。在图15中,可以假设最后非零变换系数存在于CG 25中,并且CG 25存在于左上16×16区域中。

[0392] 如图15所示,当最后非零变换系数存在于CG 25中时,由于解码设备从存在最后非零变换系数的CG开始按照逆扫描顺序扫描CG,因此解码设备基于图15中指示的数字按照25、24、23、……、3、2、1的顺序来进行扫描。由于CG 24、23、22、21、20、17、16、15、11位于左上16×16区域之外,因此仅通过如表6所示检查LastSignificantCoeffX和LastSignificantCoeffY是否对应于左上16×16区域,无法知道非零系数是否仅存在于左上16×16区域中。

[0393] 因此,在执行残差编码的解析时,如果每次扫描非零变换系数时检查对应的4×4系数组是否属于左上16×16区域,则可以仅在非零系数存在于左上16×16区域之外的情况下将MtsZeroOutSigCoeffFlag的值设置为0。在这种情况下,残差编码语法如下表所示。

[0394] [表7]

[0395]

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	描述符
if(((sps_mts_enabled_flag && cu_sbt_flag && log2TbWidth < 6 && log2TbHeight < 6)) && cIdx = = 0 && log2TbWidth > 4)	
log2ZoTbWidth = 4	
else	
log2ZoTbWidth = Min(log2TbWidth, 5)	
if((sps_mts_enabled_flag && cu_sbt_flag && log2TbWidth < 6 && log2TbHeight < 6)) && cIdx = = 0 && log2TbHeight > 4)	
log2ZoTbHeight = 4	
else	
log2ZoTbHeight = Min(log2TbHeight, 5)	
if(log2TbWidth > 0)	
last_sig_coeff_x_prefix	ae(v)
if(log2TbHeight > 0)	
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
log2TbWidth = log2ZoTbWidth	
log2TbHeight = log2ZoTbHeight	
remBinsPass1 = ((1 << (log2TbWidth + log2TbHeight)) * 7) >> 2	
log2SbW = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
log2SbH = log2SbW	
if(log2TbWidth + log2TbHeight > 3) {	
if(log2TbWidth < 2) {	
log2SbW = log2TbWidth	
log2SbH = 4 − log2SbW	
} else if(log2TbHeight < 2) {	
log2SbH = log2TbHeight	
log2SbW = 4 − log2SbH	
}	
}	

[0396]

numSbCoeff = 1 << (log2SbW + log2SbH)	
lastScanPos = numSbCoeff	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − (log2SbW + log2SbH))) − 1	
do {	
if(lastScanPos == 0) {	
lastScanPos = numSbCoeff	
lastSubBlock− −	
}	
lastScanPos− −	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][lastSubBlock][1]	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][1]	
} while((xC != LastSignificantCoeffX) || (yC != LastSignificantCoeffY))	
if(lastSubBlock == 0 && log2TbWidth >= 2 && log2TbHeight >= 2 && !transform_skip_flag[x0][y0][cIdx] && lastScanPos > 0)	
LfnstDcOnly = 0	
if((lastSubBlock > 0 && log2TbWidth >= 2 && log2TbHeight >= 2) || (lastScanPos > 7 && (log2TbWidth = = 2 || log2TbWidth = = 3) && log2TbWidth = = log2TbHeight))	
LfnstZeroOutSigCoeffFlag = 0	
if((LastSignificantCoeffX > 15 || LastSignificantCoeffY > 15) && cIdx = = 0)	
MtsZeroOutSigCoeffFlag = 0	
QState = 0	
for(i = lastSubBlock; i >= 0; i− −) {	
startQStateSb = QState	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][1]	
inferSbDcSigCoeffFlag = 0	
if((i < lastSubBlock) && (i > 0)) {	
coded_sub_block_flag[xS][yS]	ac(v)
inferSbDcSigCoeffFlag = 1	
}	

[0397]

firstSigScanPosSb = numSbCoeff	
lastSigScanPosSb = −1	
firstPosMode0 = (i == lastSubBlock ? lastScanPos : numSbCoeff − 1)	
firstPosMode1 = −1	
for(n = firstPosMode0; n >= 0 && remBinsPass1 >= 4; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag) && (xC != LastSignificantCoeffX || yC != Last SignificantCoeffY)) {	
sig_coeff_flag[xC][yC]	ae(v)
remBinsPass1− −	
if(sig_cocff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
if((xC > 15 || yC > 15) && cIdx == 0)	
MtsZeroOutSigCoeffFlag = 0	
abs_level_gtx_flag[n][0]	ae(v)
remBinsPass1− −	
if(abs_level_gtx_flag[n][0]) {	
par_level_flag[n]	ae(v)
remBinsPass1− −	
abs_level_gtx_flag[n][1]	ae(v)
remBinsPass1− −	
}	
if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	
AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gtx_flag[n][0] + 2 * abs_level_gtx_flag[n][1]	
if(pic_dep_quant_enabled_flag)	
QState = QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1]	
if(remBinsPass1 < 4)	
firstPosMode1 = n − 1	
}	
……	
}	
}	

[0398] 如表7所示,每当找到非零变换系数时,可以检查该非零变换系数是否位于左上16×16区域中(if((xC>15||yC>15)&&cIdx==0)),由此将值MtsZeroOutSigCoeffFlag设置为0。这里,当作为指示变换系数是否存在的标志信息的sig_coeff_flag[xC][yC]的值为1(if(sig_coeff_flag[xC][yC]))时检查该条件,其中xC和yC分别表示变换块内以样本为单位的x坐标和y坐标。变换块的左上位置为(0,0)。

[0399] 表7包括表6中呈现的条件检查和MtsZeroOutSigCoeffFlag的更新(if((LastSignificantCoeffX>15||LastSignificantCoeffY>15)&&cIdx==0)MtsZeroOutSigCoeffFlag=0)。如果最后非零变换系数的sig_coeff_flag[xC][yC]的值被

推断为1,则可以从表7中去除条件检查和MtsZeroOutSigCoeffFlag的更新。

[0400] 综上所述,当通过表7所示的语法配置发信号通知表4中的MTS索引时,无论应用哪种一次变换,仅当非零变换系数存在于亮度变换块的左上16×16区域中时,才发信号通知MTS索引。当不发信号通知MTS索引时,MTS索引的值被推断为0,即,应用(DCT-2,DCT-2)。

[0401] 如上所述,在以下三种情况下,可以在发信号通知MTS索引之前可以知道所应用的一次变换。

[0402] 1)在应用子块变换(SBT)的情况下

[0403] 2)在应用帧内子分区(ISP)的情况下

[0404] 3)在应用隐式MTS的情况下

[0405] 在情况(1)中,由于MTS索引被配置为仅在表4中指示是否应用SBT的cu_sbt_flag的值为0时才被发信号通知,也就是说,由于仅在不应用SBT时才发信号通知MTS索引,因此应用SBT的情况与表7的修改无关。

[0406] 在情况(2)中,由于在表4中仅在不应用ISP时才发信号通知MTS索引(在表4中当发信号通知MTS索引时检查条件(if IntraSubPartitionsSplit[x0][y0]==ISP_NO_SPLIT),因此应用ISP的情况也与表7的修改无关。

[0407] 在情况(3)中,由于在表4中仅在启用显式MTS时才发信号通知MTS索引,因此应用隐式MTS的情况也与表7的修改无关。也就是说,由于仅在应用于通过帧内预测所生成的残差块的MTS(帧内MTS)中sps_explicit_mts_intra_enabled_flag的值为1以及应用于通过帧间预测所生成的残差块的MTS(帧间MTS)中sps_explicit_mts_inter_enabled_flag的值为1时才发信号通知MTS索引,因此表7的修改对应用隐式MTS的情况没有影响。

[0408] 根据另一示例,如下表所示,可以以CG为单位检查非零系数是否存在于左上16×16区域之外。

[0409] [表8]

[0410]

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	描述符
if(((sps_mts_enabled_flag && cu_sbt_flag && log2TbWidth < 6 && log2TbHeight < 6)) && cIdx = = 0 && log2TbWidth > 4)	
log2ZoTbWidth = 4	
else	
log2ZoTbWidth = Min(log2TbWidth, 5)	
if((sps_mts_enabled_flag && cu_sbt_flag && log2TbWidth < 6 && log2TbHeight < 6)) && cIdx = = 0 && log2TbHeight > 4)	
log2ZoTbHeight = 4	
else	
log2ZoTbHeight = Min(log2TbHeight, 5)	
if(log2TbWidth > 0)	
last_sig_coeff_x_prefix	ae(v)
if(log2TbHeight > 0)	
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
log2TbWidth = log2ZoTbWidth	
log2TbHeight = log2ZoTbHeight	
remBinsPass1 = ((1 << (log2TbWidth + log2TbHeight)) * 7) >> 2	
log2SbW = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
log2SbH = log2SbW	
if(log2TbWidth + log2TbHeight > 3) {	
if(log2TbWidth < 2) {	
log2SbW = log2TbWidth	
log2SbH = 4 − log2SbW	
} else if(log2TbHeight < 2) {	
log2SbH = log2TbHeight	
log2SbW = 4 − log2SbH	
}	
}	

numSbCoeff = 1 << (log2SbW + log2SbH)	
lastScanPos = numSbCoeff	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − (log2SbW + log2SbH))) − 1	
do {	
if(lastScanPos == 0) {	
lastScanPos = numSbCoeff	
lastSubBlock− −	
}	
lastScanPos− −	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][lastSubBlock][1]	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][1]	
} while((xC != LastSignificantCoeffX) || (yC != LastSignificantCoeffY))	
if(lastSubBlock == 0 && log2TbWidth >= 2 && log2TbHeight >= 2 && !transform_skip_flag[x0][y0][cIdx] && lastScanPos > 0)	
LfnstDcOnly = 0	
[0411] if((lastSubBlock > 0 && log2TbWidth >= 2 && log2TbHeight >= 2) || (lastScanPos > 7 && (log2TbWidth = = 2 || log2TbWidth = = 3) && log2TbWidth = = log2TbHeight))	
LfnstZeroOutSigCoeffFlag = 0	
if((LastSignificantCoeffX > 15 || LastSignificantCoeffY > 15) && cIdx = = 0)	
MtsZeroOutSigCoeffFlag = 0	
QState = 0	
for(i = lastSubBlock; i >= 0; i− −) {	
startQStateSb = QState	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][1]	
inferSbDcSigCoeffFlag = 0	
if((i < lastSubBlock) && (i > 0)) {	
coded_sub_block_flag[xS][yS]	ae(v)
inferSbDcSigCoeffFlag = 1	
}	
if(coded_sub_block_flag[xS][yS] && ((xS << log2SbW) > 15 || (yS << log2SbH) > 15) && cIdx == 0)	
MtsZeroOutSigCoeffFlag = 0	

[0412]

firstSigScanPosSb = numSbCoeff	
lastSigScanPosSb = −1	
firstPosMode0 = (i == lastSubBlock ? lastScanPos : numSbCoeff − 1)	
firstPosMode1 = −1	
for(n = firstPosMode0; n >= 0 && remBinsPass1 >= 4; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag) && (xC != LastSignificantCoeffX || yC != LastSignificantCoeffY)) {	
sig_coeff_flag[xC][yC]	ae(v)
remBinsPass1− −	
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
abs_level_gtx_flag[n][0]	ae(v)
remBinsPass1− −	
if(abs_level_gtx_flag[n][0]) {	
par_level_flag[n]	ae(v)
remBinsPass1− −	
abs_level_gtx_flag[n][1]	ae(v)
remBinsPass1− −	
}	
if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	
AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gtx_flag[n][0] + 2 * abs_level_gtx_flag[n][1]	
if(pic_dep_quant_enabled_flag)	
QState = QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1]	
if(remBinsPass1 < 4)	
firstPosMode1 = n − 1	
}	
……	
}	
}	

[0413] 在表8中,xS和yS表示当前变换块内部的每个CG单元中的x坐标位置和y坐标位置,并且需要转换为以样本为单位的坐标,以检查变换系数是否位于左上16×16区域之外。也就是说,xS和yS被转换为以样本为单位的坐标,诸如(xS<<log2SbW)和(yS<<log2SbH)。这里,log2SbW和log2SbH分别表示对应CG的宽度和高度的以2为底的对数值,在变换块中x坐标从左到右增加,y坐标从上到下增加。变换块的左上位置的坐标为(0,0)。

[0414] 如表8所示,在以CG为单位进行扫描时,当每个CG内部存在非零变换系数(当coded_sub_block_flag[xS][yS]的值为1时)并且CG被识别为位于变换块的左上16×16区域之外((xS<<log2SbW)>15||(yS<<log2SbH)>15))时,MtsZeroOutSigCoeffFlag的值被设置为0。

[0415] 如表4所示,由于变量MtsZeroOutSigCoeffFlag的值被初始化为1,所以除非

MtsZeroOutSigCoeffFlag的值被设置为0,否则MtsZeroOutSigCoeffFlag的值保持为1。当对于最后非零变换系数所属的CG的coded_sub_block_flag[xS][yS]的值被推断为1时,可以从表8中去除表6中的条件检查部分和MtsZeroOutSigCoeffFlag的更新(if((LastSignificantCoeffX>15||LastSignificantCoeffY)>15)&&cIdx==0)MtsZeroOutSigCoeffFlag=0)。

[0416] 提供以下附图来描述本公开的具体示例。由于附图中例示的装置的特定术语或信号/消息/字段的特定术语是为了示例而提供的,因此本公开的技术特征不限于以下附图中使用的特定术语。

[0417] 图16是例示根据本公开的实施方式的视频解码设备的操作的流程图。

[0418] 图16中公开的每个过程是基于参照图5至图15描述的一些细节的。因此,将省略或将示意性地进行与参照图3以及图5至图15所描述的内容重复的具体细节的描述。

[0419] 根据实施方式的解码设备300可以接收包括残差信息的比特流,并且可以从比特流中推导当前块(即,要变换的变换块)的残差信息(例如,量化变换系数)(S1610)。

[0420] 具体地,解码设备300可以从比特流中解码关于当前块的量化变换系数的信息,并且可以基于关于当前块的量化变换系数的信息来推导目标块的量化变换系数。关于目标块的量化变换系数的信息可以包括在序列参数集(SPS)或条带报头中,并且可以包括关于是否应用RST的信息、关于缩减因子的信息、关于应用RST的最小变换大小的信息、关于用于应用RST的最大变换大小的信息、逆RST大小、以及关于指示变换集中包括的变换核矩阵中的任何一个的变换索引的信息中的至少一个。

[0421] 解码设备300可以基于残差信息推导当前块中最后有效系数的位置和当前块的变换系数(S1620)。解码设备300可以通过对当前块的量化变换系数进行解量化来推导变换系数。

[0422] 推导出的变换系数可以二维地布置在当前块中,并且解码设备可以通过该残差编码推导关于当前块中的非零数据(即,非零有效系数)的信息。也就是说,解码设备可以识别关于当前块中的非零有效系数的最后位置信息。

[0423] S1620中基于残差信息推导出的变换系数可以是如上所述的解量化变换系数,或者可以是量化变换系数。也就是说,变换系数仅需要是用于识别当前块中的非零数据和有效系数的位置的数据即可,而与量化无关。

[0424] 解码设备可以通过将LFNST应用于变换系数来推导修改的变换系数,或者可以不执行LFNST。

[0425] 解码设备可以通过执行变换系数或修改的变换系数的逆一次变换来推导残差样本。解码设备可以使用通用DCT-2作为变换核,或者可以应用上述MTS进行一次逆变换。

[0426] MTS可以隐式地执行,或者可以基于MTS索引的显式信令来执行。

[0427] 解码设备可以检查解析MTS索引的预定条件。根据示例,解码设备可以推导指示在当前块的除了左上第一区域之外的第二区域中是否存在有效系数的标志信息,其中可以通过以扫描有效系数的扫描子块为单位确定在第二区域中是否存在有效系数来推导标志信息(S1630)。

[0428] 根据示例,当MTS被应用于逆一次变换时,解码设备可以对要变换的当前块的左上第一区域中的变换系数执行逆一次变换。也就是说,当前块的除了左上第一区域之外的第

二区域被清零,因此不包括变换系数。

[0429] 根据示例,当应用MTS时,第一区域可以是当前块的左上16×16区域,在此情况下中第二区域可以是除了左上16×16区域之外的区域。

[0430] 扫描子块可以是4×4块,并且可以从当前块中最后有效系数的位置开始按照逆对角扫描方向来进行扫描,如图15所示。

[0431] 为了在确定在清零区域中是否存在有效系数之后解析MTS索引,解码设备可以推导指示在第二区域中是否存在有效系数的标志信息,并且可以基于标志信息来解析MTS索引。

[0432] 标志信息可以是变量MtsZeroOutSigCoeffFlag,并且可以以扫描有效系数的扫描子块为单位来确定在第二区域中是否存在有效系数。

[0433] 当表8所示的指示在扫描子块中是否存在有效系数的标志(coded_sub_block_flag或sb_coded_flag)等于1并且在扫描子块中存在的有效系数位于第二区域中(((xS<<log2SbW)>15||(yS<<log2SbH)>15))时,变量MtsZeroOutSigCoeffFlag可以指示在第二区域中存在有效系数。

[0434] 另选地,如在表7中,可以通过在每次扫描变换系数时检查4×4扫描子块是否属于第一区域来推导变量MtsZeroOutSigCoeffFlag。也就是说,当指示在扫描子块中是否存在有效系数的标志(sig_coeff_flag)等于值1并且在扫描子块中存在的有效系数位于第二区域中(if((xC>15||yC>15))时,变量MtsZeroOutSigCoeffFlag可以指示在第二区域中存在有效系数。

[0435] 另选地,当指示在扫描子块中是否存在有效系数的标志等于值1并且扫描子块位于第二区域中时,变量MtsZeroOutSigCoeffFlag可以指示在第二区域中存在有效系数。

[0436] 变量MtsZeroOutSigCoeffFlag可以初始地设置为1,当有效系数不位于第二区域中时可以保持为1,并且当有效系数位于第二区域中时可以被改变为0。

[0437] 解码设备可以基于标志信息指示在第二区域中不存在有效系数而从比特流中解析MTS索引(S1640)。

[0438] 也就是说,当初始被设置为1的变量MtsZeroOutSigCoeffFlag的值被保持时,可以解析MTS索引。

[0439] 解码设备可以通过将基于MTS索引推导出的变换核应用于第一区域中的变换系数来推导当前块的残差样本(S1650)。

[0440] 随后,解码设备300可以基于当前块的残差样本和当前块的预测样本生成重构样本。

[0441] 提供以下附图来描述本公开的具体示例。由于附图中例示的装置的特定术语或信号/消息/字段的特定术语是为了例示而提供的,因此本公开的技术特征不限于以下附图中使用的特定术语。

[0442] 图17是例示根据本公开的实施方式的视频编码设备的操作的流程图。

[0443] 图17中公开的每个过程是基于参照图5至图15描述的一些细节。因此,将省略或将示意性地进行与参照图2和图5至图15描述的内容重叠的具体细节的描述。

[0444] 根据实施方式的编码设备200可以基于应用于当前块的帧内预测模式来推导预测样本(S1710)。

[0445] 根据实施方式的编码设备200可以基于预测样本推导当前块的残差样本(S1720)。

[0446] 根据实施方式的编码设备200可以基于残差样本的一次变换来推导目标块的变换系数(S1730)。

[0447] 根据示例,编码设备可以使用通用的DCT-2作为变换核,或者可以将上述MTS应用于一次变换。

[0448] MTS可以隐式地执行,或者可以基于MTS索引的显式信令来执行。

[0449] 编码设备可以确定是否针对一次变换执行MTS,并且当确定要执行MTS时,可以通过将DCT-8或DST-7应用于残差样本来推导变换系数。

[0450] 根据示例,当针对一次变换应用MTS时,编码设备可以将当前块的除了左上第一区域之外的第二区域进行清零(S1740)。

[0451] 根据示例,第一区域可以是当前块的左上16×16区域,在这种情况下,第二区域可以是除了左上16×16区域之外的区域。

[0452] 由于这种清零,可以减少执行整个变换过程所需的计算量并减少整个变换过程所需的运算量,因此降低执行变换所需的功耗。此外,可以减少变换过程中所需要的延迟,因此增加图像编码效率。

[0453] 编码设备可以构造图像信息以按照扫描有效系数的扫描子块为单位来推导指示在第二区域中是否存在有效系数的标志信息(S1750)。

[0454] 也就是说,编码设备可以构造图像信息以使得在解码设备中可以解析表7或表8所示的图像信息。

[0455] 根据示例,扫描子块可以是4×4块,并且在解码设备中也可以从当前块中的最后有效系数的位置开始根据逆对角扫描方向来对扫描子块进行扫描,如图15所示。

[0456] 标志信息可以是变量MtsZeroOutSigCoeffFlag,并且可以以扫描有效系数的扫描子块为单位来确定在第二区域中是否存在有效系数。

[0457] 当表8所示的指示在扫描子块中是否存在有效系数的标志(coded_sub_block_flag)等于1并且在扫描子块中存在的有效系数位于第二区域中(((xS<<log2SbW)>15||(yS<<log2SbH)>15))时,变量MtsZeroOutSigCoeffFlag可以指示在第二区域中存在有效系数。

[0458] 另选地,如表7所示,可以通过在每次扫描变换系数时检查4×4扫描子块是否属于第一区域来推导变量MtsZeroOutSigCoeffFlag。也就是说,当指示在扫描子块中是否存在有效系数的标志(sig_coeff_flag)等于值1并且在扫描子块中存在的有效系数位于第二区域中(if((xC>15||yC>15))时,变量MtsZeroOutSigCoeffFlag可以指示在第二区域中存在有效系数。

[0459] 另选地,当指示在扫描子块中是否存在有效系数的标志等于值1并且扫描子块位于第二区域中时,变量MtsZeroOutSigCoeffFlag可以指示在第二区域中存在有效系数。

[0460] 变量MtsZeroOutSigCoeffFlag可以初始被设置为1,当有效系数不位于第二区域中时可以保持为1,并且当有效系数位于第二区域中时可以被改变为0。

[0461] 编码设备可以通过进一步将LFNST应用于在一次变换之后推导的变换系数来推导修改的变换系数。

[0462] 编码设备可以基于标志信息指示在第二区域中不存在有效系数,对MTS索引以及基于当前块的变换系数或修改的变换系数推导的残差信息进行编码(S1760)。

[0463] 也就是说,编码设备可以生成包括关于量化变换系数的信息的残差信息。残差信息可以包括与前述变换相关的信息/语法元素。编码设备可以对包括残差信息的图像/视频信息进行编码,并且可以以比特流的形式输出经编码的图像/视频信息。

[0464] 具体地,编码设备200可以生成关于量化变换系数的信息并且可以对所生成的关于量化变换系数的信息进行编码。

[0465] 在本公开中,可以省略量化/解量化和/或变换/逆变换中的至少一者。当省略量化/解量化时,可以将量化变换系数称为变换系数。当省略变换/逆变换时,变换系数可以被称为系数或残差系数,或者为了表述一致性仍可以被称为变换系数。

[0466] 另外,在本公开中,量化变换系数和变换系数可以分别被称为变换系数和缩放变换系数。在此情况下,残差信息可以包括关于变换系数的信息,并且可以通过残差编码语法发信号通知关于变换系数的信息。可以基于残差信息(或关于变换系数的信息)来推导变换系数,并且可以通过变换系数的逆变换(缩放)来推导缩放变换系数。可以基于缩放变换系数的逆变换(变换)来推导残差样本。这些细节也可以在本公开的其它部分中应用/表达。

[0467] 在上述实施方式中,借助于一系列步骤或方框基于流程图解释了方法,但是本公开不限于步骤的顺序,并且可以按与上述顺序或步骤不同的顺序或步骤来执行某一步骤,或某一步骤与其它步骤并发地执行。此外,本领域普通技术人员可以理解,流程图中所示的步骤不是排它的,并且在不影响本公开的范围的情况下,可以并入另一步骤或者可以删除流程图中的一个或更多个步骤。

[0468] 根据本公开的上述方法可以被实现为软件形式,并且根据本公开的编码设备和/或解码设备可以被包括在诸如电视、计算机、智能电话、机顶盒和显示装置等之类的用于图像处理的设备中。

[0469] 当本公开中的实施方式通过软件来实施时,上述方法可以被实施为用于执行上述功能的模块(步骤、功能等)。这些模块可以存储在存储器中并且可以由处理器执行。存储器可以在处理器内部或外部,并且可以以各种众所周知的方式连接到处理器。处理器可以包括专用集成电路(ASIC)、其它芯片组、逻辑电路和/或数据处理装置。存储器可以包括只读存储器(ROM)、随机存取存储器(RAM)、闪存、存储卡、存储介质和/或其它存储装置。也就是说,本公开中描述的实施方式可以在处理器、微处理器、控制器或芯片上实施和执行。例如,每个附图中所示的功能单元可以在计算机、处理器、微处理器、控制器或芯片上实施和执行。

[0470] 此外,应用本公开的解码设备和编码设备可以包括在多媒体广播收发器、移动通信终端、家庭影院视频装置、数字影院视频装置、监控相机、视频聊天装置、实时通信装置(诸如视频通信)、移动流装置、存储介质、摄像机、视频点播(VoD)服务提供装置、顶置(OTT)视频装置、互联网流服务提供装置、三维(3D)视频装置、视频电话视频装置、和医疗视频装置,并且可以用于处理视频信号或数据信号。例如,顶置(OTT)视频装置可以包括游戏机、蓝光播放器、互联网接入TV、家庭影院系统、智能电话、平板PC、数字视频记录仪(DVR)等。

[0471] 另外,应用于本公开的处理方法可以以由计算机执行的程序的形式来生产,并且可以存储在计算机可读记录介质中。具有根据本公开的数据结构的多媒体数据也可以存储在计算机可读记录介质中。计算机可读记录介质包括存储计算机可读数据的各种存储装置和分布式存储装置。计算机可读记录介质可以包括例如蓝光盘(BD)、通用串行总线(USB)、

ROM、PROM、EPROM、EEPROM、RAM、CD-ROM、磁带、软盘和光学数据存储装置。此外,计算机可读记录介质包括以载波(例如,互联网上的传输)形式实施的介质。另外,通过编码方法所生成的比特流可以存储在计算机可读记录介质中,或者通过有线或无线通信网络来传输。另外,本公开的实施方式可以通过程序代码被实施为计算机程序产品,并且程序代码可以按照本公开的实施方式在计算机上执行。程序代码可以存储在计算机可读载体上。

[0472] 本文所公开的权利要求可以以各种方式组合。例如,本公开的方法权利要求的技术特征可以被组合以在设备中实现或执行,并且设备权利要求的技术特征可以被组合以在方法中实现或执行。此外,方法权利要求和设备权利要求的技术特征可以被组合以在设备中实现或执行,并且方法权利要求和设备权利要求的技术特征可以被组合以在方法中实现或执行。

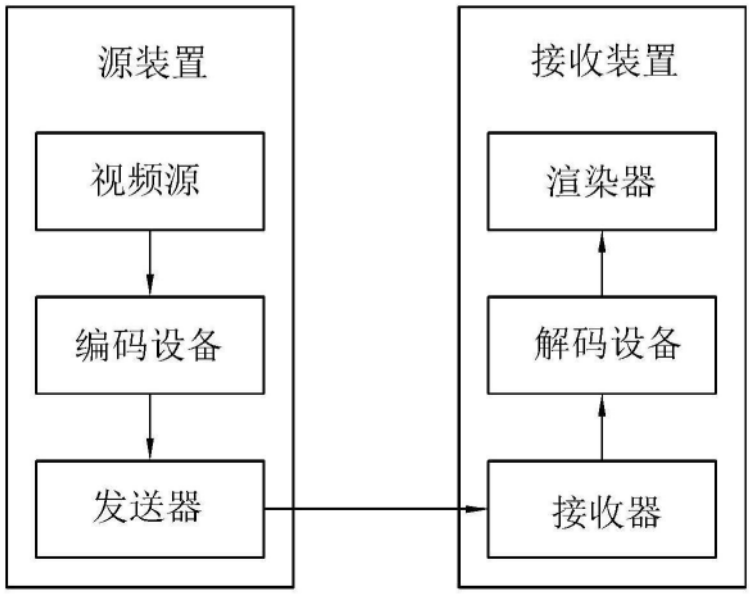


图1

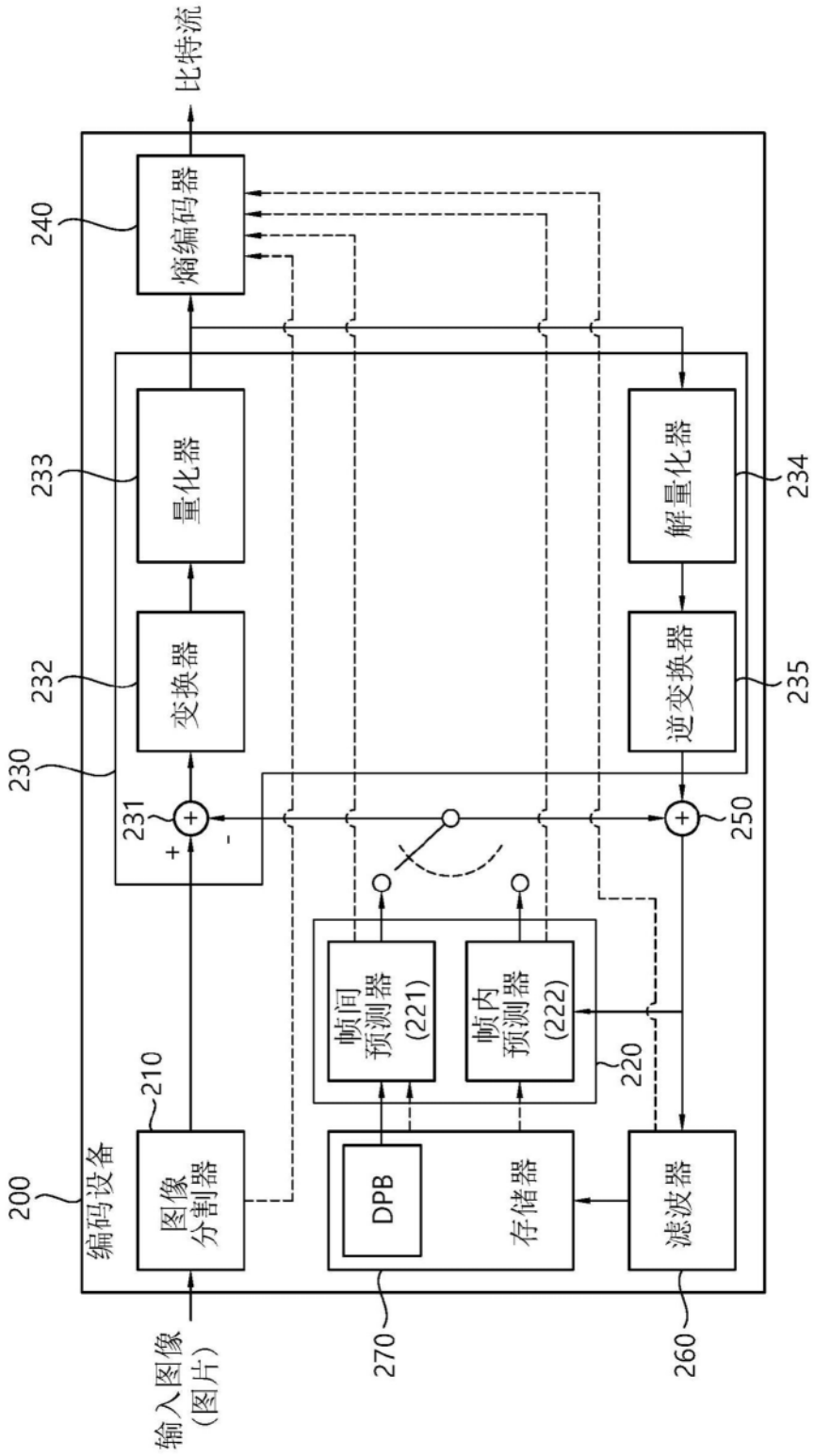

比特流
熵编码器
240
量化器
233
变换器
232
230
231
+
-
解量化器
234
逆变换器
235
250
帧间
预测器
(221)
帧内
预测器
(222)
220
210
图像
分割器
编码设备
200
DPB
存储器
滤波器
输入图像
(图片)
270
260

图2

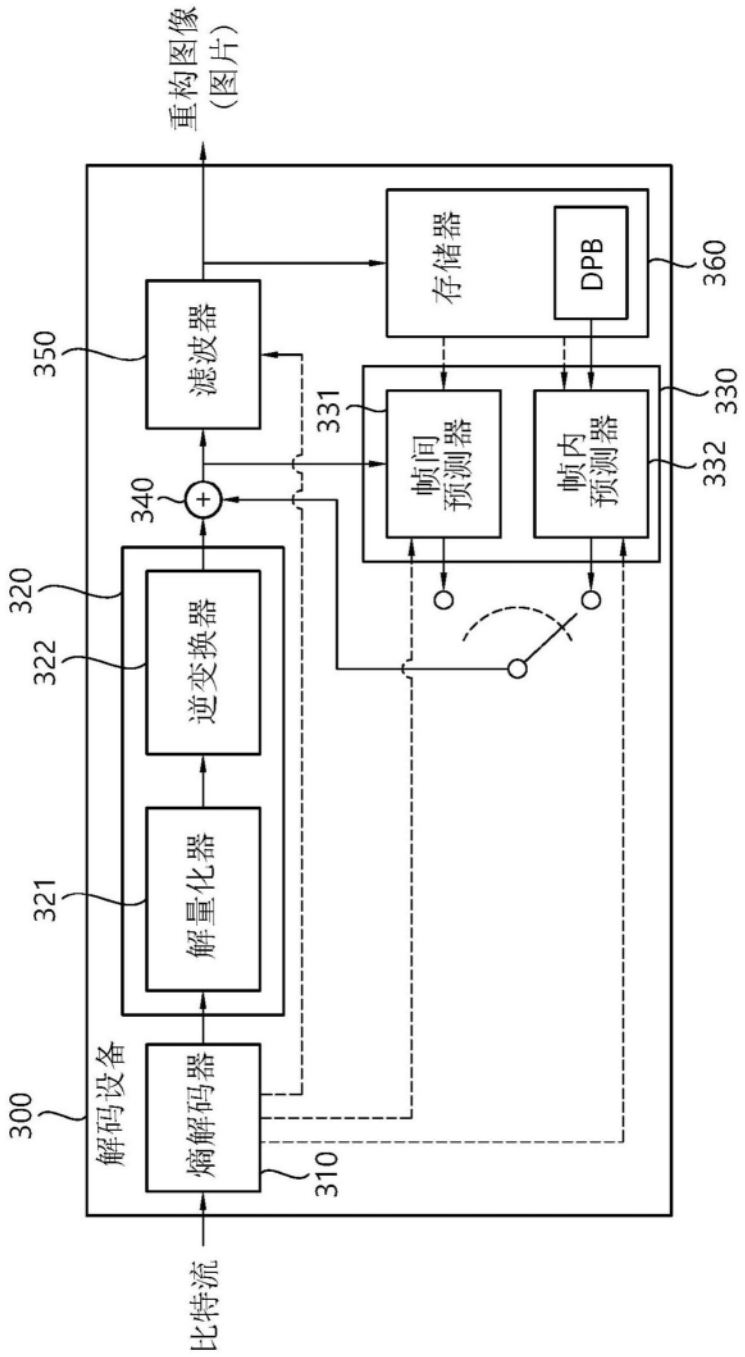

重构图像
(图片)
存储器
DPB
360
滤波器
350
331
帧间
预测器
帧内
预测器
330
332
340
320
322
逆变换器
321
解量化器
300
解码设备
熵解码器
310
比特流

图3

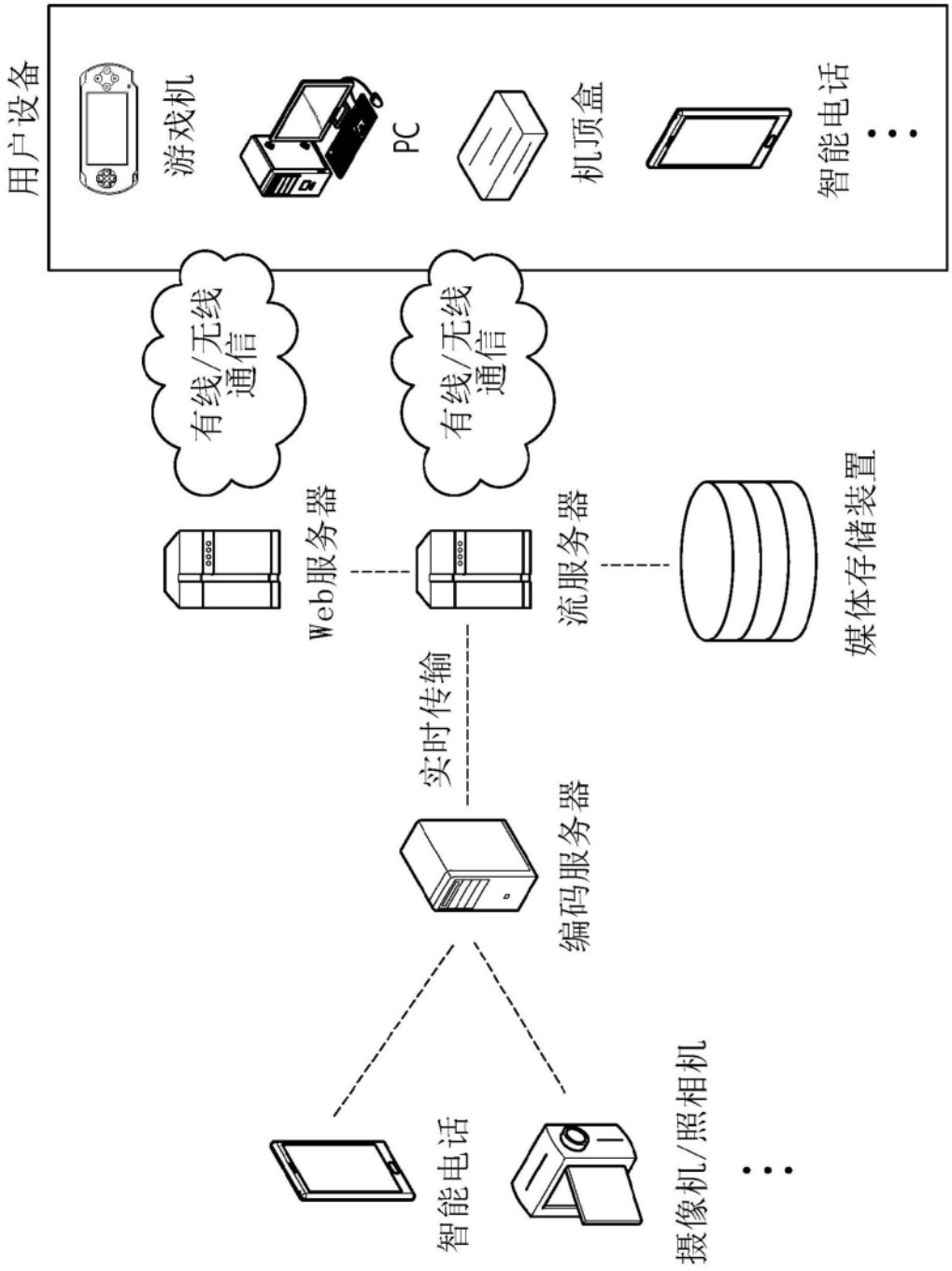

用户设备
游戏机
PC
机顶盒
智能电话
...
有线/无线通信
有线/无线通信
Web服务器
流服务器
媒体存储装置
实时传输
编码服务器
智能电话
摄像机/照相机
...

图4

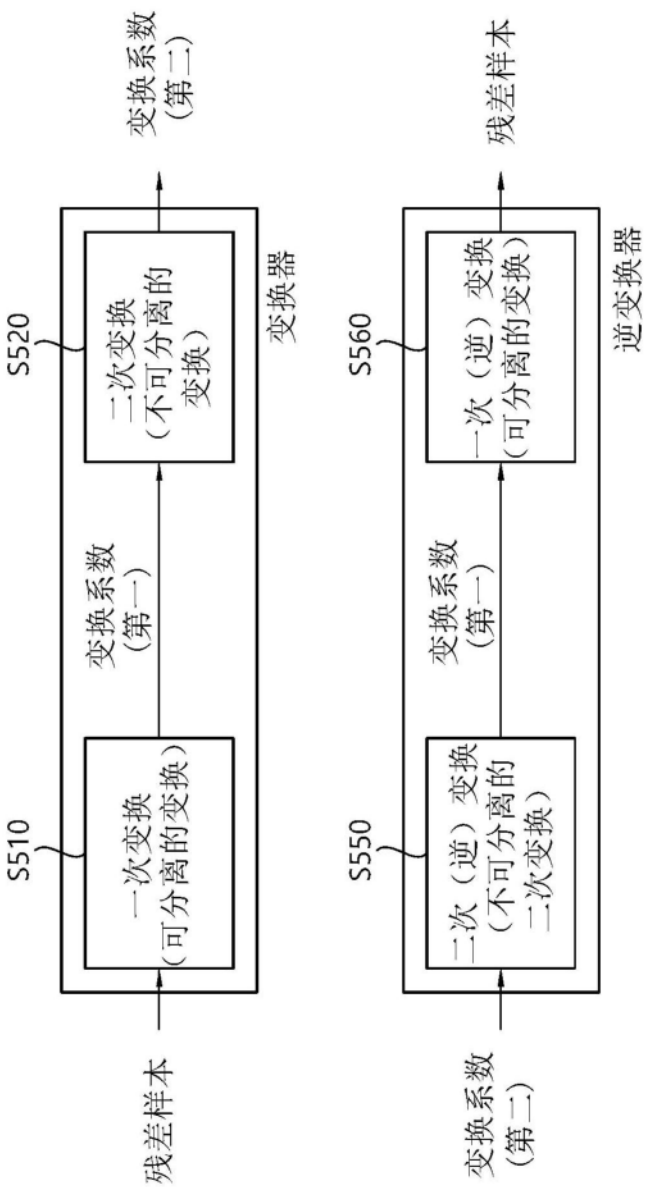

残差样本
S510
一次变换
（可分离的变换）
变换系数
（第一）
S520
二次变换
（不可分离的
变换）
变换系数
（第二）
变换器
变换系数
（第二）
S550
二次（逆）变换
（不可分离的
二次变换）
变换系数
（第一）
S560
一次（逆）变换
（可分离的变换）
残差样本
逆变换器

图5

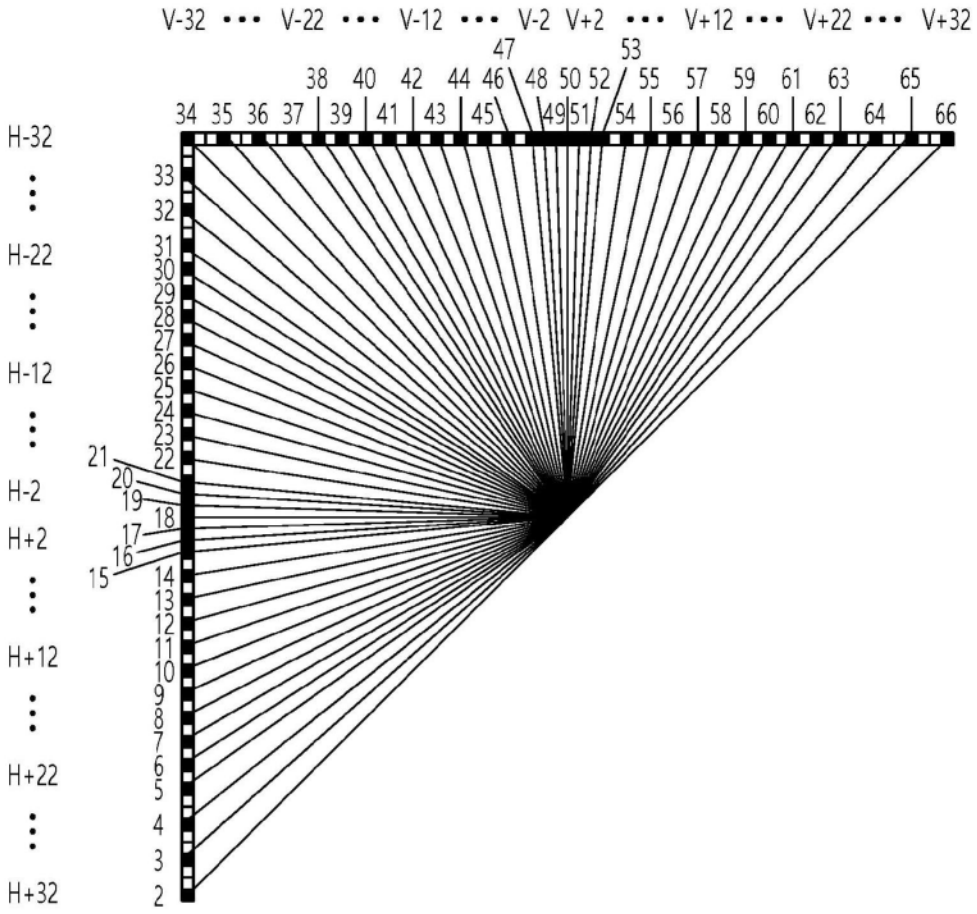


图6

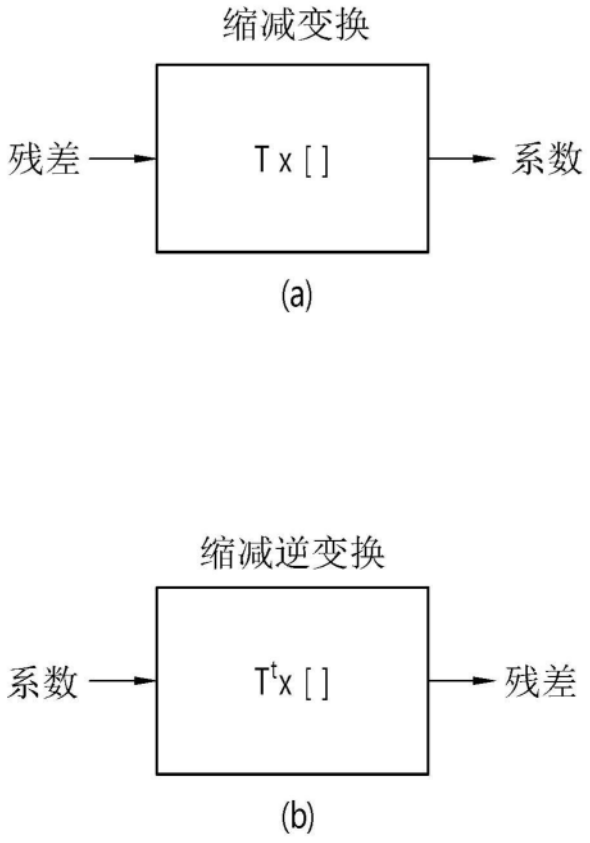


图7

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36				
37	38	39	40				
41	42	43	44				
45	46	47	48				

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

(a)

1	9	17	25	33	37	41	45
2	10	18	26	34	38	42	46
3	11	19	27	35	39	43	47
4	12	20	28	36	40	44	48
5	13	21	29				
6	14	22	30				
7	15	23	31				
8	16	24	32				

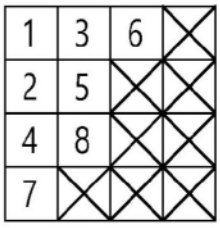
1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

(b)

图8

1	3	6	10
2	5	9	13
4	8	12	15
7	11	14	16

(a)



(b)

图9

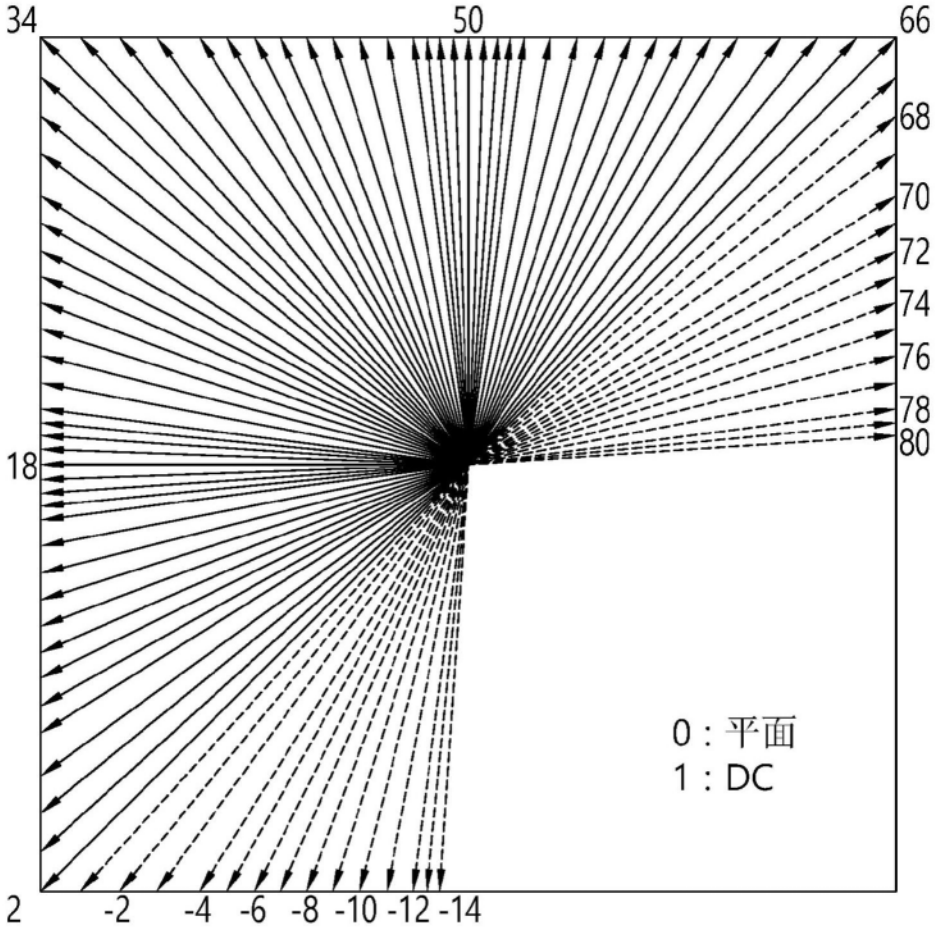
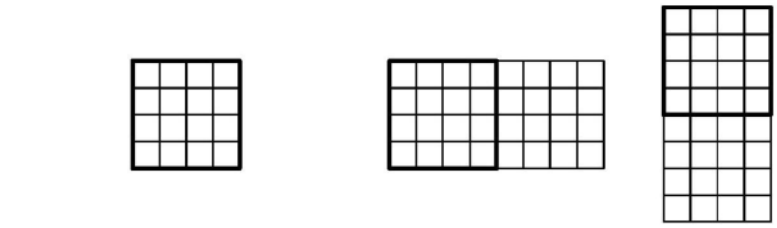
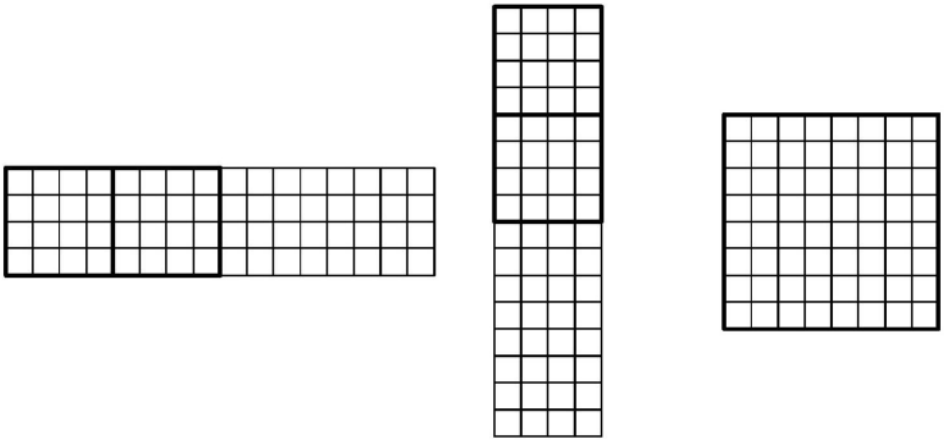


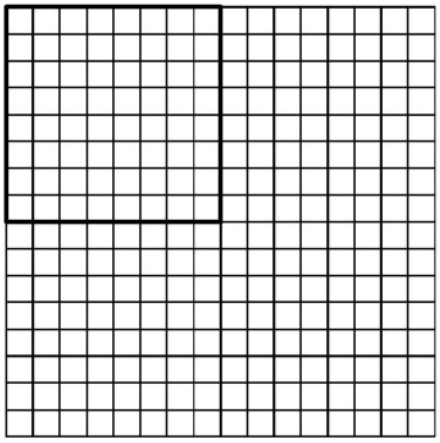
图10



(a) 4x4 (b) 8x4 / 4x8



(c) 4xN / Nx4, 当N≥16时 (d) 8x8



(e) MxN(M≥8, N≥8, M>8或N>8)

图11

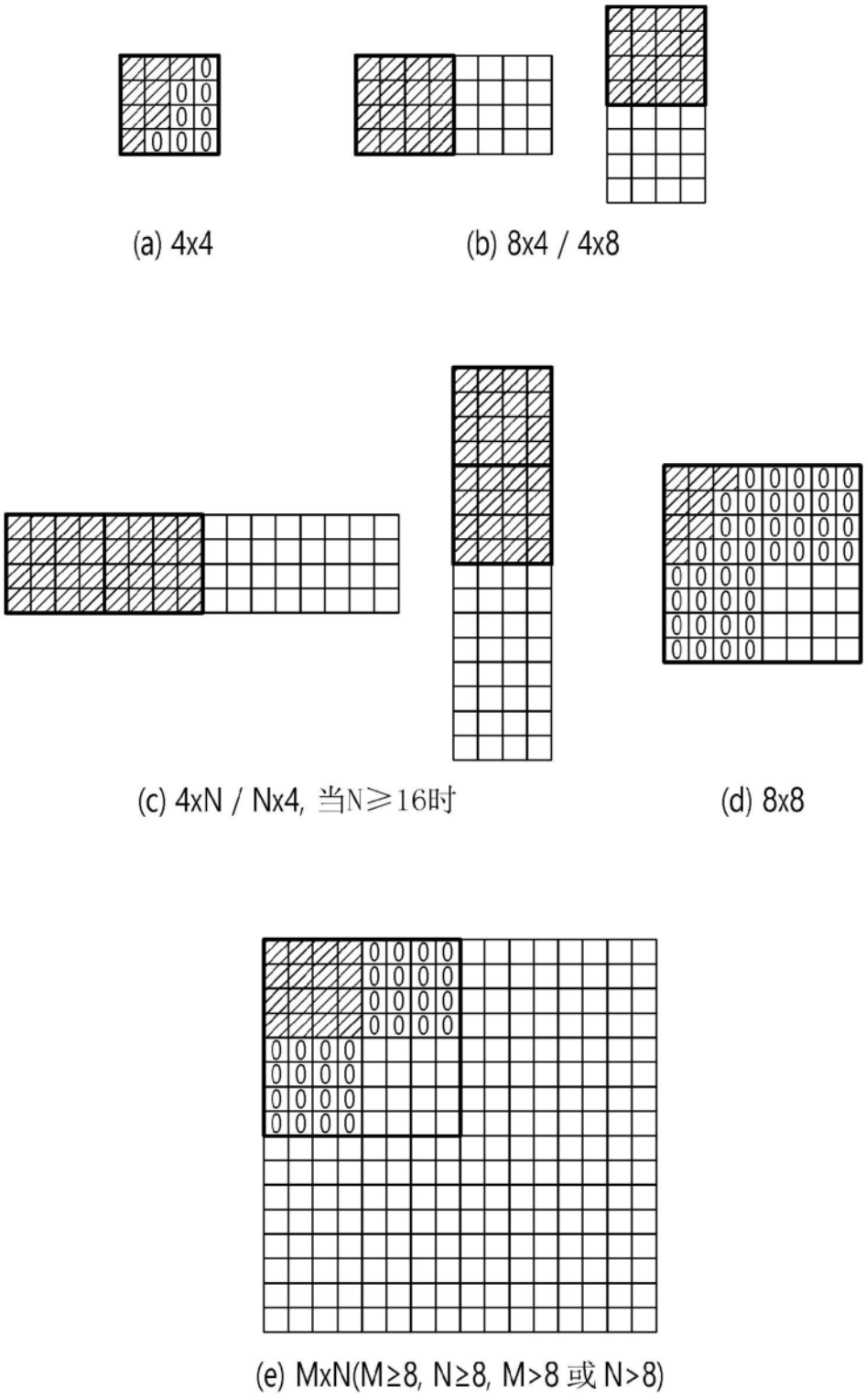
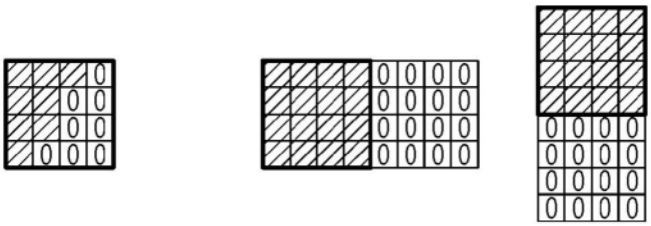
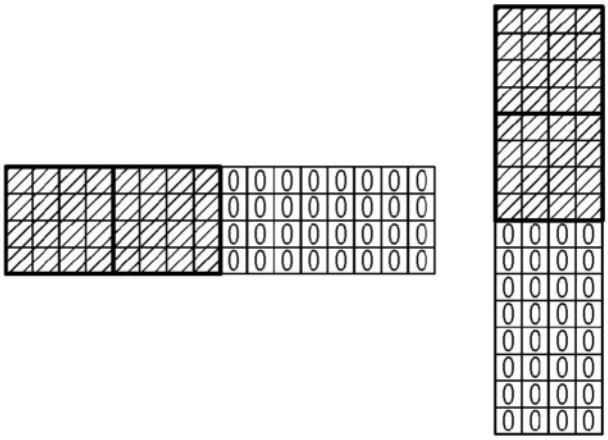


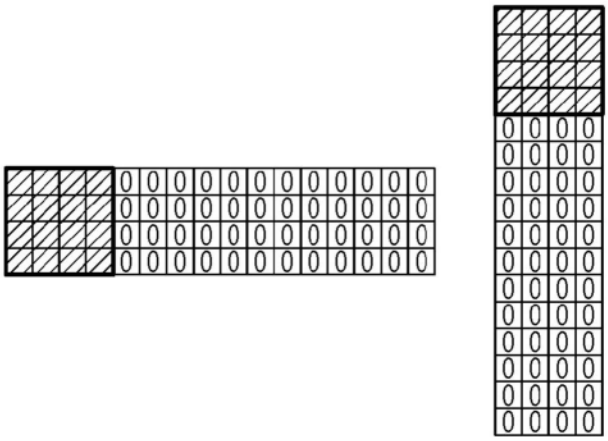
图12



(a) 4x4 (b) 8x4 / 4x8



(c) 4xN / Nx4, 当N≥16时



(d) 4xN / Nx4, 当N≥16时

图13

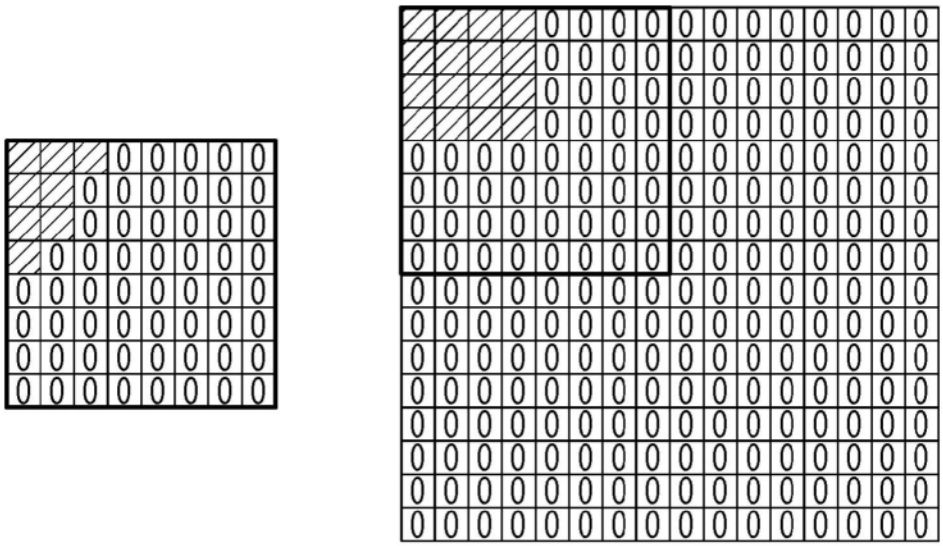


图14

1	3	6	10	15	21	28	36
2	5	9	14	20	27	35	43
4	8	13	19	26	34	42	49
7	12	18	25	33	41	48	54
11	17	24	32	40	47	53	58
16	23	31	39	46	52	57	61
22	30	38	45	51	56	60	63
29	37	44	50	55	59	62	64

图15

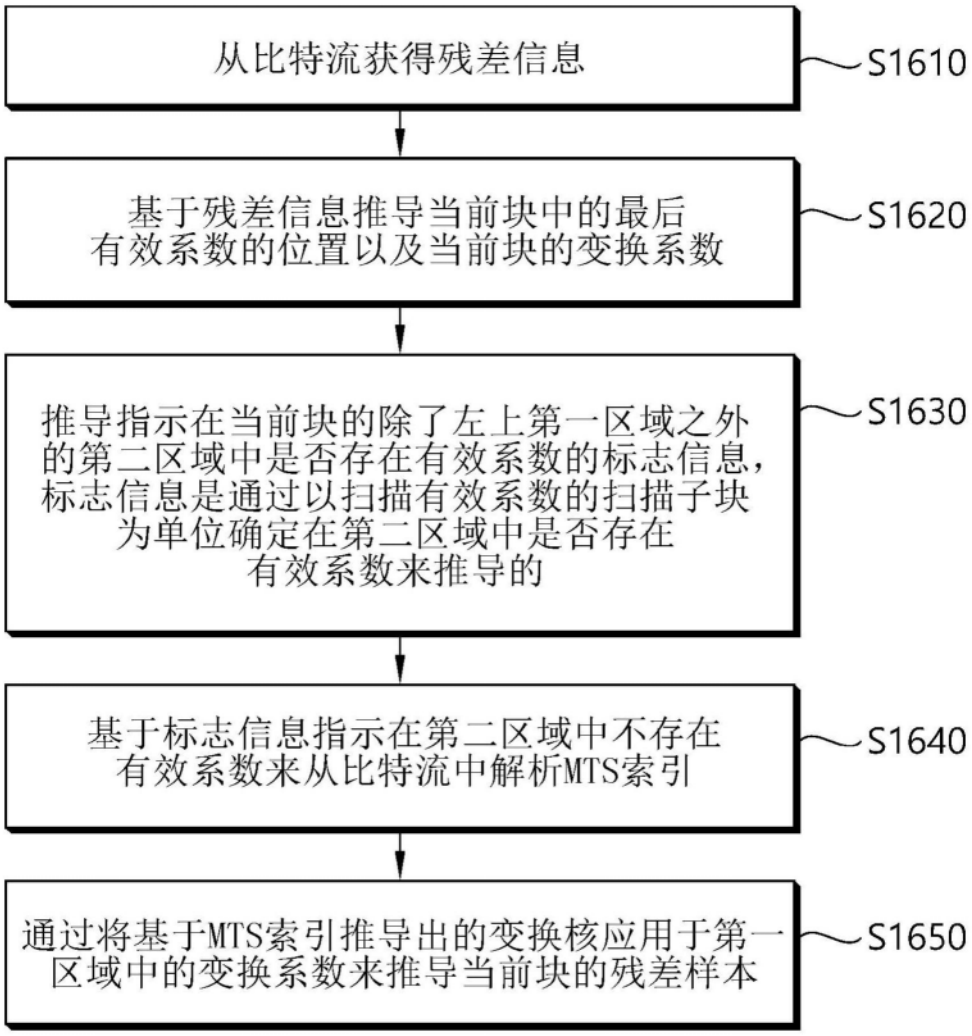

从比特流获得残差信息
S1610
基于残差信息推导当前块中的最后有效系数的位置以及当前块的变换系数
S1620
推导指示在当前块的除了左上第一区域之外的第二区域中是否存在有效系数的标志信息，标志信息是通过以扫描有效系数的扫描子块为单位确定在第二区域中是否存在有效系数来推导的
S1630
基于标志信息指示在第二区域中不存在有效系数来从比特流中解析MTS索引
S1640
通过将基于MTS索引推导出的变换核应用于第一区域中的变换系数来推导当前块的残差样本
S1650

图16

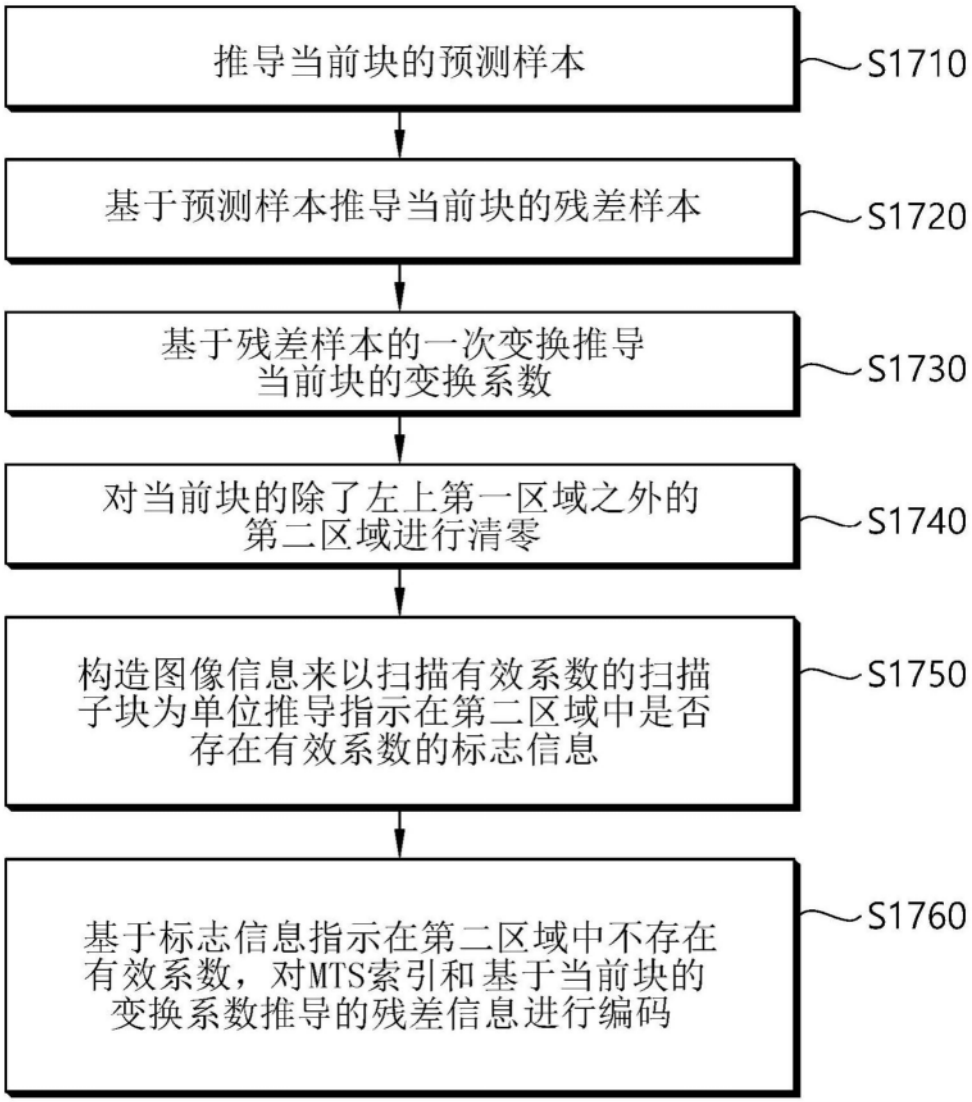

推导当前块的预测样本
S1710
基于预测样本推导当前块的残差样本
S1720
基于残差样本的一次变换推导
当前块的变换系数
S1730
对当前块的除了左上第一区域之外的
第二区域进行清零
S1740
构造图像信息来以扫描有效系数的扫描
子块为单位推导指示在第二区域中是否
存在有效系数的标志信息
S1750
基于标志信息指示在第二区域中不存在
有效系数，对MTS索引和基于当前块的
变换系数推导的残差信息进行编码
S1760

图17